

|  |
| --- |
| LTP for CCSDS Interoperability Testing |

DRAFT CCSDS Record

CCSDS 000.0-Y-0

Draft Yellow Book

January 2015

FOREWORD

[Foreword text specific to this document goes here. The text below is boilerplate.]

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This document is therefore subject to CCSDS document management and change control procedures, which are defined in the *Procedures Manual for the Consultative Committee for Space Data Systems*. Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be addressed to the CCSDS Secretariat at the address indicated on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

* Agenzia Spaziale Italiana (ASI)/Italy.
* Canadian Space Agency (CSA)/Canada.
* Centre National d’Etudes Spatiales (CNES)/France.
* China National Space Administration (CNSA)/People’s Republic of China.
* Deutsches Zentrum für Luft- und Raumfahrt (DLR)/Germany.
* European Space Agency (ESA)/Europe.
* Federal Space Agency (FSA)/Russian Federation.
* Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
* Japan Aerospace Exploration Agency (JAXA)/Japan.
* National Aeronautics and Space Administration (NASA)/USA.
* UK Space Agency/United Kingdom.

Observer Agencies

* Austrian Space Agency (ASA)/Austria.
* Belgian Federal Science Policy Office (BFSPO)/Belgium.
* Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
* China Satellite Launch and Tracking Control General, Beijing Institute of Tracking and Telecommunications Technology (CLTC/BITTT)/China.
* Chinese Academy of Sciences (CAS)/China.
* Chinese Academy of Space Technology (CAST)/China.
* Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
* Danish National Space Center (DNSC)/Denmark.
* Departamento de Ciência e Tecnologia Aeroespacial (DCTA)/Brazil.
* European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
* European Telecommunications Satellite Organization (EUTELSAT)/Europe.
* Geo-Informatics and Space Technology Development Agency (GISTDA)/Thailand.
* Hellenic National Space Committee (HNSC)/Greece.
* Indian Space Research Organization (ISRO)/India.
* Institute of Space Research (IKI)/Russian Federation.
* KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
* Korea Aerospace Research Institute (KARI)/Korea.
* Ministry of Communications (MOC)/Israel.
* National Institute of Information and Communications Technology (NICT)/Japan.
* National Oceanic and Atmospheric Administration (NOAA)/USA.
* National Space Agency of the Republic of Kazakhstan (NSARK)/Kazakhstan.
* National Space Organization (NSPO)/Chinese Taipei.
* Naval Center for Space Technology (NCST)/USA.
* Scientific and Technological Research Council of Turkey (TUBITAK)/Turkey.
* South African National Space Agency (SANSA)/Republic of South Africa.
* Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
* Swedish Space Corporation (SSC)/Sweden.
* Swiss Space Office (SSO)/Switzerland.
* United States Geological Survey (USGS)/USA.

DOCUMENT CONTROL

|  |  |  |  |
| --- | --- | --- | --- |
| **Document** | **Title and Issue** | **Date** | **Status** |
| CCSDS 000.0-Y-0 | LTP FOR CCSDS INTEROPERABILITY TESTING Draft CCSDS Record, Issue 0 | January 2015 | Current draft |
|  |  |  |  |
|  |  |  |  |

CONTENTS

Section Page

[DOCUMENT CONTROL iii](#_Toc402433415)

[CONTENTS iv](#_Toc402433416)

[1 Introduction 1-1](#_Toc402433417)

[1.1 Purpose 1-1](#_Toc402433418)

[1.2 Scope 1-1](#_Toc402433419)

[1.3 Applicability 1-1](#_Toc402433420)

[1.4 References 1-1](#_Toc402433421)

[2 Overview 3](#_Toc402433422)

[2.1 Licklider Transmission Protocol (LTP) for CCSDS 3](#_Toc402433423)

[2.2 Test Infrastructure 4](#_Toc402433424)

[3 Implementations 6](#_Toc402433425)

[4 Requirements from Section 6 of the LTP for CCSDS Recommended Standard 4-1](#_Toc402433426)

[5 Interoperability Tests 5-1](#_Toc402433427)

[6 Test Results 6-1](#_Toc402433428)

[6.1 Summary 6-1](#_Toc402433429)

[7 Issues Identified During Testing 7-2](#_Toc402433430)

[7.1 Sender Response to RS from Closed Sessions 7-2](#_Toc402433431)

[7.2 SessionOriginator in SessionID of Report Segments Should Be What? 7-2](#_Toc402433432)

[7.3 Offsets of ReceptionClaims 7-2](#_Toc402433433)

[7.4 HMAC/SHA1 Authentication 7-2](#_Toc402433434)

[7.5 Authentication Trailer Length 7-2](#_Toc402433435)

[7.6 Authentication Key Material 7-3](#_Toc402433436)

[7.7 Mixed-Color Reception in ION 7-3](#_Toc402433437)

[7.7.1 Fix 7-3](#_Toc402433438)

[7.8 Mixed-Color Transmission in ION 7-3](#_Toc402433439)

[7.8.1 Fix 7-3](#_Toc402433440)

[7.9 Receiving Red Segment in Green Part of Block 7-3](#_Toc402433441)

[7.10 Service Data Aggregation 7-4](#_Toc402433442)

# Introduction

## Purpose

This report documents the interoperability testing conducted in support of the LTP for CCSDS Recommended Standard. This document supports the CCSDS requirement that a Recommended Standard be accompanied by a report documenting interoperability between two independent prototypes or implementations.

## Scope

This document applies to the LTP for CCSDS specification only.

## Applicability

This document supports the interoperability testing requirement needed for the LTP for CCSDS Specification to be accepted by CCSDS as a Recommended Standard.

## References

The following publications are referenced in this document. At the time of publication, the editions indicated were valid. All publications are subject to revision, and users of this document are encouraged to investigate the possibility of applying the most recent editions of the publications indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS publications.

[1] Licklider Transmission Protocol (LTP) for CCSDS. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 734.1-R-1. Washington, D.C.: CCSDS, XXXXX 2014

[2] M. Ramadas, S. Burleigh, and S. Farrell. *Licklider Transmission Protocol—Specification*. RFC 5326. Reston, Virginia: ISOC, September 2008.

[3] S. Farrell, M. Ramadas, and S. Burleigh. *Licklider Transmission Protocol—Security Extensions*. RFC 5327. Reston, Virginia: ISOC, September 2008.

[4] *Encapsulation Service*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.1-B-2. Washington, D.C.: CCSDS, October 2009.

[5] *Space Packet Protocol*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.0-B-1. Washington, D.C.: CCSDS, September 2003.

[6] [www.secdev.org/projects/scapy/](http://www.secdev.org/projects/scapy/)

# Overview

## Licklider Transmission Protocol (LTP) for CCSDS

The Licklider Transmission Protocol (LTP) [1, 2] provides a data transmission service to move blocks of data from one LTP engine to another, where in general the two engines are resident in separate data systems, often with a single connecting space link.

Each block consists logically of two parts, either of which may be of length zero. The first part, termed the ‘red-part’, is transmitted reliably between LTP entities, using acknowledgements and retransmissions to ensure that the entire red-part is received correctly at the receiver; this provides a reliable transmission service. The second part of the block, termed the ‘green-part’, consists of data to be transmitted unreliably. Data in the green-part is not subject to acknowledgements and retransmissions and therefore provides an unreliable service. The LTP client Service Instance controls what data in a block is ‘red’ and what is ‘green’. A Client Service Instance that desires completely reliable data transfer must therefore specify that all of the data be sent as ‘red’ (reliable) data. In this specification, the ability to send/receive green-part data is optional. However, if green-part capability is supported, then both transmission and reception must be supported. Block transmission may span periods of disconnection. During these periods, retransmission timers maintained by LTP are suspended.

Figure 2-1 illustrates the LTP transmission process.



Figure ‑1 : Overview of LTP Interactions

In addition to the base data transmission protocol, the LTP for CCSDS specification allows for the use of a subset of the security mechanisms described in the security extensions specification [3].

## LTP Segment Types

Table 2-1 summarizes the LTP segment types.

Table 2-1: LTP Segment Types

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CTRL | EXC | Flag 1 | Flag 0 | Code | Nature of segment |
| 0 | 0 | 0 | 0 | 0 | Red data, NOT {Checkpoint, EORP or EOB} |
| 0 | 0 | 0 | 1 | 1 | Red data, Checkpoint, NOT {EORP or EOB} |
| 0 | 0 | 1 | 0 | 2 | Red data, Checkpoint, EORP, NOT EOB |
| 0 | 0 | 1 | 1 | 3 | Red data, Checkpoint, EORP, EOB |
|  |  |  |  |  |  |
| 0 | 1 | 0 | 0 | 4 | Green data, NOT EOB |
| 0 | 1 | 0 | 1 | 5 | Green data, undefined |
| 0 | 1 | 1 | 0 | 6 | Green data, undefined |
| 0 | 1 | 1 | 1 | 7 | Green data, EOB |
|  |  |  |  |  |  |
| 1 | 0 | 0 | 0 | 8 | Report segment |
| 1 | 0 | 0 | 1 | 9 | Report-acknowledgment segment |
| 1 | 0 | 1 | 0 | 10 | Control segment, undefined |
| 1 | 0 | 1 | 1 | 11 | Control segment, undefined |
|  |  |  |  |  |  |
| 1 | 1 | 0 | 0 | 12 | Cancel segment from block sender |
| 1 | 1 | 0 | 1 | 13 | Cancel-acknowledgment segment to block sender |
|  |  |  |  |  |  |
| 1 | 1 | 1 | 0 | 14 | Cancel segment from block receiver |
| 1 | 1 | 1 | 1 | 15 | Cancel-acknowledgment segment to block receiver |

## LTP Cancel Reason Codes

Table 2 summaries the LTP Cancel Reason Codes

Table 2-2: LTP Cancel Reason Codes

|  |  |  |
| --- | --- | --- |
| **Value** | **Mnemonic** | **Description** |
| 0x00 | USR\_CNCLD | Client service canceled session. |
| 0x01 | UNREACH | Unreachable client service. |
| 0x02 | RLEXC | Retransmission limit exceeded. |
| 0x03 | MISCOLORED | Received either a red-part data segment at block offset above any green-part data segment offset or a green-part data segment at block offset below any red-part data segment offset. |
| 0x04 | SYS\_CNCLD | A system error condition caused unexpected session termination. |
| 0x05 | RXMTCYCEXC | Exceeded the Retransmission-Cycles limit. |
| 0x06 – 0xFF | Reserved |  |

## Test Infrastructure

The test infrastructure uses Linux Containers (Linux Network Namespaces) to segregate the two implementations under test.

Each instance locally communicates with a UDP-to-CCSDS-over-XXX gateway to convert between LTP over UDP and LTP over CCSDS Encapsulation Service [4] / CCSDS Space Packets Service [5] over UDP.

The network connection between the two implementations under test is via a bridge in the Linux host. This allows the imposition of Linux traffic control (TC) measures such as the network emulation (netem) module to emulate delay and loss.

The network connection between the two LTP implementations imposed an additional delay, generally 500ms in the tests, and for some of the tests also imposed packet losses. These constitute the simulated operationally relevant environment required by the CCSD procedures manual

For some tests, one or more host processes monitored the test process in order to modify the communications parameters. This allowed the test to drive the implementations into particular conditions. For example, a monitoring process might use ebtables to block communications after a checkpoint segment has been sent but before the report segment is received. This would cause the sender to retransmit the checkpoint segment until communications were re-established.



Figure 2-2: Overview of test configuration.

The test infrastructure is implemented in Python. Each individual test is described in a Python test script with support from an overall infrastructure script. The steps in test execution are:

1. Make a test directory and copy the relevant files into it
2. Set up the virtual containers and wait for them to stabilize
3. Configure the network connectivity (bridge) between them, including any latency and loss
4. Start packet captures on the two virtual Ethernet interfaces in the host
5. Start any monitoring process(es) needed by the test
6. Execute the LTP data transfer by invoking the ‘test’ operation of the test script.
7. Stop the packet captures
8. Post-process the packet captures for possible visualization
9. Analyze the test results by invoking the ‘analyze’ operation of the test script

The analyses rely heavily on the Python ‘scapy’ package [6] augmented to understand the LTP for CCSDS protocol, as well as the CCSDS Encapsulation Packet [5] and CCSDS Space Packet [6] formats.

## Implementations

The two implementations used for this testing were:

1. The ION LTP implementation, derived from ion-open-source-3.2.0 with TopCoder modifications to implement authentication per the LTP for CCSDS specification.
2. A Python-based implementation written by Keith Scott of the MITRE Corporation.

## Summary Conclusions

The tests demonstrated interoperability between the two independently-developed implementations of the LTP for CCSDS specification [1]. Issues identified as part of the interoperability testing are documented in section 5 . The largest issue had to do with the format of the LTP Service Data Aggregation capability, and the specification was modified to adhere to the ION LTP implementation of the capability because it provides more flexibility and is more efficient than was described in the original specification.

All of the required capabilities from section 6 of the LTP for CCSDS specification were tested in at least one direction, and most were exercised in both directions. Here ‘direction’ is an LTP session from and LTP sender to and LTP receiver, which may involve bi-directional communication. The rationale for testing some of the capabilities in only one direction is that some capabilities, especially those dealing with error conditions, were difficult to invoke for one or the other of the implementations under test. For example, the Python implementation has a command-line option to inject LTP Headers and Trailers that are unknown to the ION implementation, which can then be verified to ignore them. No such switch exists for the ION LTP implementation.

# Requirements from Section 6 of the LTP for CCSDS Recommended Standard

## Overview

Table 3-1 lists the requirements from section 6 of the LTP for CCSDS Recommended Standard. In many cases the ‘Requirement Text’ in the table is the applicable requirement text from the LTP for CCSDS document, however in some cases it has been shortened to fit here.

These are the criteria against which interoperability is measured. For each of the requirements in the table, at least one test was constructed to demonstrate the requirement. Note that not all of the requirements in the table relate to interoperability. For example, the service interface notifications, while part of section 6 of the LTP for CCSDS document, are not applicable to the ‘on-the-wire’ interoperability between implementations.

For each test, the numbers of the interoperability tests that verify the requirement in each direction are listed. Descriptions of the interoperability tests are in the section 4.

## LTP for CCSDS Requirements

Table 3-1: LTP for CCSDS Requirements

| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Successful ION->Python Tests | Successful Python->ION Tests |
| --- | --- | --- | --- | --- |
| 6.2.1.1 | RFC5326 3.0 | **Segment structure** -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. | 1-5, 7-8, 10, 13-32, 34-34 | 1-9, 11, 13-22, 24-33 |
| 6.2.1.1 | RFC5326 3.1(.0) | **Segment Header** consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | 1-5, 7-8, 10, 13-32, 34-34 | 1-9, 11, 13-22, 24-33 |
| 6.2.1.1 | RFC5326 3.1.4 | **Extensions Field Format (in header)** -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) | 24-30 | 1, 24-30 |
| 6.2.1.1 | RFC5326 3.1.4 | **Extensions Format** -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] | 24-30 | 24-30 |
| 6.2.1.1 | RFC5326 3.2.1 | **Data Segment Format --** Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. | 1-5, 7-8, 10, 13-32, 34-34 | 1-9, 11, 13-22, 24-33 |
| 6.2.1.1 | RFC5326 3.2.2 | **Report Segment Format --** RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report | 1-3, 5, 8, 13-14, 17-19, 21-22, 24, 26-27, 29-30, 32, 34-34 | 1-3, 5-6, 8, 13-14, 17-18, 21-22, 24, 26-27, 29-30, 32-33 |
| 6.2.1.1 | RFC5326 3.2.3 | **Report-Acknowledgement Segment Format -- RA segments** contain a report serial number. | 1-3, 5, 13-14, 17-19, 21-22, 24, 27, 30, 34-34 | 1-3, 5-6, 13-14, 17-18, 21-22, 24, 27, 30-30 |
| 6.2.1.1 | RFC5326 3.2.4 | **Cancel segment reason codes** | 8, 10, 15-20, 23, 25-26, 28-29, 31-32, 34-34 | 8-9, 11, 15-16, 19-20, 25-26, 28-29, 31-33 |
| 6.2.1.1 | RFC5326 3.3 | **Segment Trailer** contains extensions encoded as TLVs | 24-30 | 1, 24-30 |
| 6.2.1.1 | RFC5326 4.1 | **Transmission.request primitive** | 1-5, 7-8, 10, 13-32, 34-34 | 1-9, 11-22, 24-33 |
| 6.2.1.1 | RFC5326 4.2 | **Cancellation.request primitive** | 10, 23 | 9, 11-12 |
| 6.2.1.1 | RFC5326 6.0 | ***Overriding Rule*** *1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further* | 14 | 14 |
| 6.2.1.1 | RFC5326 6.0 | ***Overriding Rule*** *2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment* | 16 | 16 |
| 6.2.1.1 | RFC5326 6.0 | ***Overriding Rule*** *3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH* | 15 | 15 |
| 6.2.1.1 | RFC5326 6.1 | **Start Transmission** -- Receipt of link cue that transmission can begin results in starting transmission. | 1-5, 7-8, 10, 13-32, 34 | 1-9, 11, 13-22, 24-33 |
| 6.2.1.1 | RFC5326 6.2 | **Start Checkpoint Timer** -- When a checkpoint is transmitted, start checkpoint timer. | 1-3, 5, 7-8, 13-14, 16-20, 22-32, 34 | 1-3, 5-8, 13-14, 16-22, 24-33 |
| 6.2.1.1 | RFC5326 6.2 | **Start Checkpoint Timer** -- Immediately suspend timer if it is known that the receiver is not transmitting. | 7-8 | 7-8 |
| 6.2.1.1 | RFC5326 6.3 | **Start Report Segment Timer** -- When a report segment is transmitted, start the report segment timer. | 13, 26, 29, 32, 34 | 3, 6, 13, 26, 29, 32-33 |
| 6.2.1.1 | RFC5326 6.3 | **Start Report Segment Timer** -- Immediately suspend timer if it is known that the remote engine is not transmitting. | 8 | 8 |
| 6.2.1.1 | RFC5326 6.4 | **Stop Transmission** -- On receipt of a link cue that transmission should be stopped, stop transmitting. | 2-3 | 2-3 |
| 6.2.1.1 | RFC5326 6.5 | **Suspend Timers** -- When transmission is halted, stop countdown timers associated with the remote engine. | 2-3 | 2-3 |
| 6.2.1.1 | RFC5326 6.6 | **Resume Timers** -- When transmission is resumed, resume countdown timers associated with the remote engine. | 2-3 | 2-3 |
| 6.2.1.1 | RFC5326 6.7 | **Retransmit Checkpoint** -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. | 2-3, 8, 13, 16, 20, 23, 25-26, 28-29, 31, 34 | 2-3, 8, 13, 16, 20, 25-26, 28-29, 31, 33 |
| 6.2.1.1 | RFC5326 6.7 | **Retransmit Checkpoint** -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. | 31 | 31, 33 |
| 6.2.1.1 | RFC5326 6.8 | **Retransmit Report Segment** -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report | 13, 26, 29, 32, 34 | 3, 13, 26, 29, 32-33 |
| 6.2.1.1 | RFC5326 6.8 | **Retransmit Report Segment** -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. | 29, 32 | 29, 32 |
| 6.2.1.1 | RFC5326 6.9 | **RedPartReception.indication** | 1-3, 5, 7-8, 13-14, 17-18, 21-22, 24, 26-27, 29-30, 32 | 1-3, 5-8, 13-14, 17-18, 21, 24, 26-27, 29-30, 32 |
| 6.2.1.1 | RFC5326 6.10 | **GreenPartSegmentArrival.indication** | 4-5, 13, 20 | 4-5, 13, 17-20 |
| 6.2.1.1 | RFC5326 6.11 | **Send Reception Report** -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. | 1-3, 5, 8, 13-14, 17-19, 21-22, 24, 26-27, 29-30, 32, 34 | 1-3, 5-6, 8, 13-14, 17-18, 21-22, 24, 26-27, 29-30, 32-33 |
| 6.2.1.1 | RFC5326 6.11 | **Send Reception Report** -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. | 32, 34 | 32 |
| 6.2.1.1 | RFC5326 6.12 | **InitialTransmissionCompletion.indication** -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. | 1-5, 7-8, 10, 13-20, 22-32, 34 | 1-8, 13-14, 16-18, 20-22, 24-33 |
| 6.2.1.1 | RFC5326 6.13 | **Retransmit Data** -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. | 13 | 13 |
| 6.2.1.1 | RFC5326 6.14 | **Stop Report Segment (RS) Timer** -- stop RS timer on receipt of a report ack for the report. | 1-3, 5, 13-14, 17-19, 21-22, 24, 27, 30, 34 | 1-2, 5-6, 13-14, 17-18, 21-22, 24, 27, 30, 33 |
| 6.2.1.1 | RFC5326 6.15 | **Start Cancel Timer** -- when a cancel semgnet is sent, start cancel timer. | 8, 10, 15-16, 23, 25-26, 28-29, 31-32, 34 | 9, 32-33 |
| 6.2.1.1 | RFC5326 6.15 | **Start Cancel Timer** -- Immediately suspend if known that the peer is not transmitting. | 8, 10, 15-16, 23, 25-26, 28-29, 32, 34 | 9, 32-33 |
| 6.2.1.1 | RFC5326 6.16 | **Retransmit Cancellation Segment** -- If the number of retransmissions exceeds the limit, cancel the session. | 16, 25-26, 28-29, 31-32 | 16, 31-32 |
| 6.2.1.1 | RFC5326 6.16 | **Retransmit Cancellation Segment** -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). | 8, 10, 15-16, 23, 25-26, 28-29, 31-32, 34 | 9, 26, 32-33 |
| 6.2.1.1 | RFC5326 6.17 | **Acknowledge Cancellation** -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. | 16 | 16 |
| 6.2.1.1 | RFC5326 6.17 | **Acknowledge Cancellation to Sender (CS)** -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. | 8, 23, 26, 29 | 8-9, 26, 29, 33 |
| 6.2.1.1 | RFC5326 6.17 | **Acknowledge Cancellation to Recevier (CR)** -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. | 10, 15, 17-20 | 15, 19-20 |
| 6.2.1.1 | RFC5326 6.18 | **Stop Cancel Timer** -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. | 10, 15, 17-20, 23 | 9, 15, 19-20, 26, 33 |
| 6.2.1.1 | RFC5326 6.19 | **Cancel Session**: dump everything queued for transmission; stop all countdown timers. | 8, 10, 15-20, 23, 25, 28, 31, 34 | 8-9, 11-12, 15-16, 19-20, 25, 28, 31, 33 |
| 6.2.1.1 | RFC5326 6.20 | **Close Session** -- on session closure, stop any countdown timers and remove session (session no longer recognized) | 32 | 32 |
| 6.2.1.1 | RFC5326 6.21 | **Handle Miscolored Segment** -- discard segment and cancel session. | 18 | 18 |
| 6.2.1.1 | RFC5326 7.1 | **TransmissionSessionStart.indication** | 1-5, 7-8, 10, 13-20, 22-32, 34 | 1-9, 11-22, 24-33 |
| 6.2.1.1 | RFC5326 7.3 | **Red-Part Reception.indication** | 1-3, 5, 7-8, 13-14, 17-18, 21-22, 24, 26-27, 29-30, 32 | 1-3, 5-8, 13-14, 17-18, 21-22, 24, 26-27, 29-30, 32 |
| 6.2.1.1 | RFC5326 7.4 | **Transmission-Session Completion.indication** | 1-5, 13-14, 24, 27, 30, 32 | 1-6, 13-14, 17-18, 21-22, 24, 27, 30, 32 |
| 6.2.1.1 | RFC5326 7.5 | **Transmission-Session Cancellation.indication** | 8, 10, 15-20, 23, 25, 28, 31, 34 | 8-9, 11-12, 15-16, 19-20, 25-26, 28-29, 31, 33 |
| 6.2.1.1 | RFC5326 7.6 | **Reception-Session Cancellation.indication** | 8, 10, 17-20, 23, 26, 29, 32, 34 | 8-9, 11, 19-20, 26, 29, 32-33 |
| 6.2.1.1 | RFC5326 7.7 | **Initial-Transmission Completion.indication** | 1-5, 7-8, 10, 13-20, 22-32, 34 | 1-8, 13-14, 16-18, 20, 24-33 |
| 6.2.1.12 | RFC5326 7.1 | **ReceptionSessionStart.indication** -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. | 1-5, 7-8, 10, 13-14, 17-20, 24, 26-27, 29-30, 32, 34 |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | **SDA: RedPartReception.indication** -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. | 21-23 | 21-22 |
| 6.2.1.12 | CCSDS 7.2.1.1 | **SDA: Transmission.request** -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. | 21-23 | 21-22 |
| 6.2.1.12 | CCSDS 7.2.1.1 | **SDA: InitialTransmissionCompletion.indication** Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. | 21-23 | 21-22 |
| 6.2.1.12 | CCSDS 7.2.2.1 | **SDA Client Service ID is '2'** -- The client service ID passed by SDA to LTP shall be ‘2’, signifying ‘LTP Service Data Aggregation’. | 21-23 | 21-22 |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | **SDA: Entire SDA SDU passed to LTP is red** -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be ‘red’ (reliably transmitted) data. | 21-23 | 21-22 |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | **SDA: Size limit reached** -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of ‘red’ data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. | 21 | 21 |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | **SDA: Time limit reached** -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of ‘red’ data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. | 22 | 22 |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | **SDA: TransmissionSessionCompletion.indications** -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. | 22 | 22 |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | **SDA: Red-Part reception, extraction, and delivery** -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. | 21-22 | 21-22 |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | **SDA: Reception cancellation.indications** -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule’s client service ID. | 23 |  |
| 6.2.1.1 | 8.1 (red) | **State Transition diagram** |  |  |
| 6.2.1.1 | 8.2 (green) | **State Transition diagram** |  |  |
| 6.2.1.12 | CCSDS 3.8.2 | **Use IANA registry for LTP Extension identifiers** | 24-30 | 24-30 |
| 6.2.1.2 | CCSDS 3.1.2 | **LTP Authentication per RFC5327** -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. | 24-30 | 24-30 |
| 6.2.1.2 | CCSDS 3.9.1 | **LTP Authentication** -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. | 24-30 | 24-30 |
| 6.2.1.2 | CCSDS 3.9.4 | **Ciphersuite Identification** -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). | 24-30 | 24-30 |
| 6.2.1.3 | CCSDS 3.9.2 | **No Cookies** -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. | 24-30 | 24-30 |
| 6.2.1.2 | CCSDS 3.9.3 | **MIB Control of Authentication** -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. | 24-30 | 24-30 |
| 6.2.1.2 | CCSDS 3.9.5 | **MIB Control of Authentication** --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. | 24-26, 28-29 | 24-26, 28-29 |
| 6.2.1.2 | CCSDS 3.9.6 | **Authentication is 'all-or-none'** -- If authentication is used, it must be included on either all LTP segments [of a session] or none. | 24-30 | 24-30 |
| 6.2.1.2 | RFC5327 2.1 | **Authentication Structure** -- LTP Authentication extension structure and ciphersuite definitions. | 24-30 | 24-30 |
| 6.2.1.4 | CCSDS 3.3.1 | **LTP over UDP** -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. | 1-11, 13-36 | 1-11, 13-36 |
| 6.2.1.4 | CCSDS 3.3.2 | **UDP Port 1113** -- Implementations of LTP over UDP should use the ‘ltp-deepspace’ UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) | The implementations are both configurable as to which UDP ports they should use for transmission and reception. Both were configured to transmit to a UDP port other than 1113 to communicate with the UDP-to-CCSDS gateways. | |
| 6.2.1.5 | CCSDS 3.5.1 | **Session Number Selection** -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] | 1-5, 7-8, 10, 13-32, 34 | 1-9, 11, 13-22, 24-33 |
| 6.2.1.6 | CCSDS 3.5.2 | **Initial Checkpoint Serial Number Range** -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. | 1-3, 5, 7-8, 13-14, 16-32, 34 | 1-3, 5-8, 13-14, 16-22, 24-33 |
| 6.2.1.6 | CCSDS 3.5.3 | **Initial** **Checkpoint Serial Number Value** -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. | 1-3, 5, 7-8, 13-14, 16-32, 34 | 1-3, 5-8, 13-14, 16-22, 24-33 |
| 6.2.1.7 | CCSDS 3.8.1 | **Ignore unknown LTP Extensions** -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  | 1 |
| 6.2.1.9 | CCSDS A3.1 | **LTP Encapsulation in Space Packets** -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. | 3 | 3 |
| 6.2.1.9 | CCSDS A3.2 | **LTP in Space Packets APID** -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) | No such APID currently exists. An arbitrary APID was used during testing. The LTP for CCSDS specification requests the allocation of an APID to identify Space Packets carrying LTP segments. | |
| 6.2.1.8 | CCSDS A2.1 | **Encapsulation in Encap Packets** -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. | 1-2, 4-5, 7-8, 10, 13-32, 34 | 1-2, 4-9, 11, 13-22, 24-33 |
| 6.2.1.9 | CCSDS A2.2 | **Protocol Identifier for LTP in Encapsulation Packets** -- The protocol identifier for the Encapsulation Service to be used to identify Encapsulation Packets carrying LTP segments as their payloads should be that specified in the SANA Protocol Identifier for Encapsulation Service registry (reference [8]). | No such protocol identifier currently exists. An arbitrary identifier was used during testing. The LTP for CCSDS specification requests the allocation of a protocol identifier to identify Encapsulation Packets carrying LTP segments. | |
| 6.2.1.10 | CCSDS 3.5.5 | **Initial Report Serial Number Range** -- The initial report serial number values used by conformant implementations must be in the range [1, 214-1]. | 1-3, 5, 8, 13-14, 17-19, 21-22, 24, 26-27, 29-30, 32, 34 | 1-3, 5-6, 8, 13-14, 17-18, 21-22, 24, 26-27, 29-30, 32-33 |
|  | CCSDS 3.5.6 | **Initial Report Serial Number Value** -- The initial report serial number values used by conformant implementations should be chosen at random. | 1-3, 5, 8, 13-14, 17-19, 21-22, 24, 26-27, 29-30, 32, 34 | 1-3, 5-6, 8, 13-14, 17-18, 21-22, 24, 26-27, 29-30, 32-33 |
| 6.2.1.11 | RFC5326 6.1, 6.10, 7.2, portions of 8.1 and 8.2 | **Green (Unreliable) Data** -- If an implementation supports sending or receiving of green (unreliable) data, it must support both sending and receiving of green data as described in sections 6.1, 6.10, and 7.2 of RFC5326, and the portions of sections 8.1 and 8.2 of RFC5326 that pertain to green data. | 4-5, 13, 20 | 4-5, 13, 17-20 |

Untested Requirements

The following table lists the untested items and the rationale for not testing them. The working group requests CESG waivers for the testing of these items.

Table 3-2: Untested Items

| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Rationale for Not Testing |
| --- | --- | --- | --- |
| 6.2.1.1 | 8.1 (red) | State Transition diagram | The state transition diagram is a pictorial representation of the LTP protocol described in the rest of the recommendation. |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) | The implementations both RECEIVE on UDP port 1113; they transmit to a different UDP port on the UDP-to-CCSDS gateway for ease of identification of packets in the packet captures. |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) | There is not currently an APID assigned. An arbitrary APID was chosen for use during testing.  This document requests the allocation of an APID to identify Space Packets Carrying LTP segments. |
| 6.2.1.9 | CCSDS A2.2 | Protocol Identifier for LTP in Encapsulation Packets -- The protocol identifier for the Encapsulation Service to be used to identify Encapsulation Packets carrying LTP segments as their payloads should be that specified in the SANA Protocol Identifier for Encapsulation Service registry (reference [8]). | There is not currently a protocol identifier assigned. An arbitrary identifier was chosen for use during testing.  This document requests the allocation of a protocol identifier to identify CCSDS Encapsulation Packets Carrying LTP segments. |

# 

# Interoperability Tests

The following table summarizes the interoperability tests.

Table 4-1: Interoperability Tests

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Number | Test Name | Short Description | Procedure | Test-Specific Success Criteria | Comments |
|  |  |  |  |  |  |
| 1 | Basic1 | Send 100B block containing only Red (reliable) data. | Sender sends a 100-Byte Red LTP Block to a listening service on an LTP receiver. Exercises both implementations of the UDP-to-EncapOverUDP gateway to provide an interoperability test of the LTP over UDP LSA (pcaps over loopback) and LTP over Encapsulation LSA (pcaps between engines). |  |  |
| 2 | Basic1B - Space | Send a 100B block containing only Red (reliable) with link cues. Block the link from [0, 45]; suspend transmission at sender during [10, 30]. Use LTP over Space Packet Encapsulation. | Use ebtables to block the link from sender to receiver from t=0 to t=45s.  In LTP implementation, suspend transmission between t=10 and t=30 seconds [configurable in Python; use spans in ion].  Use the two different implementations of LTP over Space Packet Encapsulation to exercise the Space Packet LSA. | Because of the link cues, there should be no retransmissions during the time [10, 30]. Once the link is restored, data transfer should resume and complete successfully. |  |
| 3 | Basic2 | Send a 1024B block containing only Green (unreliable) data | Sender sends a 1024 byte LTP Block containing only Green (unreliable) data. |  |  |
| 4 | Basic3 | Block with 2kBRed and 4kBGreen | Sender sends an LTP Block containing a mix of Red and Green data. | Demonstrates use of 1- and 2-byte SDNVs in red-part segments. |  |
| 5 | Basic4 | Multi-checkpoint Red data | Send an LTP block containing Red data wehre a checkpoint is generated DURING the red-part of the data transmission, and not only at the end. | Ensure that the sender generates a "Checkpoint, not EORP, not EOB" segment (type 1) and that the receiver responds to it correctly. | ION LTP implementation doesn't support multi-checkpointing. |
| 6 | NTR | Red data to non-transmitting receiver | Sender is configured to know that receiver can't transmit when it sends the block. | Checkpoint timers at the sender should be immediately suspended (there should be asingle checkpoint transmission and no checkpoint retransmsissions). |  |
| 7 | NTS | Red data to non-transmitting sender (for the purposes of report generation). | Configure the receiver to know that the sender is not allowed to transmit (report acknowledgements). | Receiver will send ONE report before suspending report timer.  Sender may retransmit checkpoint multiple times.  Sender may cancel the session due to too many retransmissions. | This test uses ebtables to block reception of the single report generated in order to prevent the sender from responding to it (otherwise there would be rather stringent timing constraints on the link cues in order to allow the checkpoint through but not respond to the report). The report segment should be visible coming from the receiver, but will NOT be visible in the packet capture at the sender. |
| 8 | Cancel1 | User cancel at sender during red-part transmission. | After starting the transfer and while it is ongoing, drop the link (without providing any link cues to the sender) and then cancel the session at the sender (by the sending LTP user). After a short period of time (to allow for retransmission of cancellation segments) restore the link. | Cancellation segments sent by sender; cancellation segments retransmitted by sender while link down; cancellation ACK sent by receiver after link is restored; sender ceases transmission of cancellation segments after receiving cancellation ACK. No data carrying segments sent by sender after user cancel processed; a single cancel-ACK sent in response to a given cancel segment. Cancellation indications delivered to sending and receiving LTP clients. | Can fail the automated test checks if for some reason Python implementation gets behind and sends two cancel segments (that get through) back-to-back; eliciting two cancel-acks. |
| 9 | Cancel2 | User cancel at receiver during red-part data transmission; drop link to elicit retransmission of cancel segments. | After starting the transfer and while it is ongoing, drop the link (without providing any link cues to the sender) and then cancel the session at the receiver (by the receiving LTP user). After a short period of time (to allow for retransmission of cancellation segments) restore the link. | Ensure that cancellation segments are retransmitted by the receiver while the link is down; ensure that a cancellation ACK is sent by sender after the link is restored; ensure that the receiver ceases transmission of cancellation segments after receiving the cancellation ACK. No segments shold be sent by the receiver after the cancel ACK is received; ensure that a single cancel-ACK sent in response to a given cancel segment.  Cancellation indications should be delivered to sending and receiving LTP clients. | Automated check may state failure at receiver due to reception of Red data after cancel segment is issued (but while link is still down so the sender doesn't know) |
| 10 | Cancel3 | User cancel at sender with a non-transmitting receiver during red-part data transmission. That is, the sender knows that the receiver is not allowed to transmit when the session is canceled. | After starting the transfer and while it is ongoing, cancel the session at the sender (by the sending LTP user). Receiver is non-transmitting and sender knows it. | A single cancel segment sent by sender. Cancellation indication delivered to sending LTP client. |  |
| 11 | Cancel4 | User cancellation at sender before any data is sent (red-part data submitted). | After sending the transmission request but before any data can be sent, sender cancels. | No LTP segments are ever transmitted. |  |
| 12 | Error1 | 10KRed, 5KGreen, 20% packet loss rate. |  | Correct reception of the Red-Part Data; receipt of at least some Green-Part Data. |  |
| 13 | Corrupt | Send a malformed segment (e.g. bad version #) in the middle of a red data transmission. | During an LTP transfer session with only red data, send a malformed segment then complete the session. | No LTP segment is generated by the receiver in response to the malformed segment (malformed segment is silently discarded); data transfer succeeds. |  |
| 14 | Unreach1 | Sender transmitting red data to an unreachable destination client service ID but receiver has queue to sender. | Attempt to open an LTP connection to an LTP client service that is not extant at the receiver. | Receiver responds with an an LTP cancel segment with reason code: UNREACH to the sender. Sender closes the connection on receipt of the UNREACH segment. |  |
| 15 | Unreach2 | Unreachable client service ID and receiver has no queue to sender. | LTP sender sends to a receiver / client pair where there is not listening client and the receiver does not know how to transmit to the sender. |  |  |
| 16 | Miscolored1 | Red segment in Green Space | After establishing a connection and transmitting the Red Part data (including the EORP segment), and beginning to send the Green Part, send a Red segment with an offset in the Green sequence space. | Receiver responds with an LTP cancel segment with reason code: MISCOLORED (and closes the connection). Sender cancels the session on receipt of the cancellation segment and sends a cancel ACK. |  |
| 17 | Miscolored2 | Green segment in Red space | After establishing a connection and transmitting the Red Part data (including the EORP segment), send a Green segment with an offset in the Red sequence space. | Receiver responds with an LTP cancel segment with reason code: MISCOLORED (and closes the connection). Sender cancels the session on receipt of the cancellation segment. |  |
| 18 | Miscolored3 | Red segment in Green Space after blocking initial EORP. | Block the initial EORP, start sending Green segments, then put a red segment in the green space. | Receiver responds with an LTP cancel segment with reason code: MISCOLORED (and closes the connection). Sender cancels the session on receipt of the cancellation segment. |  |
| 19 | Miscolored4 | Green segment in known Red space. | Get a green segment in the KNOWN RED space and before the EORP segment. Use ebtables to block the receipt of the EORP so that the green segment shows up in the known red space before the Red part of the block is completely received. |  |  |
| 20 | SDALen | Service data aggregation(SDA) by length. | Aggregation service running at the sender and client; aggregation size at the sender set to 1,000 bytes, aggregation time set to 5 seconds.  Sending client sends 'messages' to the aggregation service at the receiver. Each message is consists of 1 octet of message length followed by (length-1) bytes of 'message'. Sender chooses the lengths of the messages to send and ensures that it sends enough messages fast enough to invoke the 1,000-byte send trigger. All messages are sent as red-part data. | Receiver correctly demultiplexes the messages provided by the sending aggregator. Aggregate messages (LTP blocks) are transmitted according to the aggregation size limit setting.  Verify that TransmissionSessionCompletion.indication indications are issued for each of the client SDUs. |  |
| 21 | SDATime | Service data aggregation (SDA) by time. | Aggregation service running at the sender and client; aggregation size at the sender set to 1,000 bytes, aggregation time set to 5 seconds.  Sending client uses the aggregation service to send a few messages, staying under the 1,000-byte aggregation size trigger. Verify that aggregated groups are send out approx. every 5s. | Receiver correctly demultiplexes the messages provided by the sending aggregator. Aggregate messages (LTP blocks) are transmitted according to the aggregation time setting.   Verify that TransmissionSessionCompletion.indication indications are issued for each of the client SDUs. |  |
| 22 | SDACancel | Cancel SDA receive session and ensure that N ReceptionSessionCancellation.indications are sent to the LTP client (1 per SDA client capsule). | Cancel an SDA session from the receiver and ensure that N (where N is the number of data capsules in the SDA aggregate) ReceptionSessionCancellation.indications to the clients. | \*\* DEFINE \*\* |  |
| 23 | Auth1 | 10KB Red data, ciphersuite 0 both directions; good keys. | Send a red block with LTP Authentication ciphersuite 0 turned on in both directions. Authenticate all segments (data, reports, acks). | Ensure that authentication is invoked and that the block is received correctly. |  |
| 24 | Auth1b | 10KB Red data, ciphersuite 0 both directions; bad sender to receiver key. | Send a red block with LTP Authentication ciphersuite 0 turned on in both directions. Authenticate all segments (data, reports, acks) but using the WRONG KEYS for the sender -> receiver direction. | Data segments rejected by receiver. |  |
| 25 | Auth1c | 10KB Red data, ciphersuite 0 both directions; bad receiver to sender key. | Send a red block with LTP Authentication ciphersuite 0 turned on in both directions. Authenticate all segments (data, reports, acks) but using the WRONG KEYS for the receiver -> sender direction. | Segments accepted by receiver; report segments rejected by the sender. |  |
| 26 | Auth2 | 10KB Red data, ciphersuite 1 both directions; good keys. | Like Auth1 but with ciphersuite 1 | Ensure that authentication is invoked and that the block is received correctly. |  |
| 27 | Auth2b | 10KB Red data, ciphersuite 1 both directions; bad sender to receiver key. | Like Auth1b but with ciphersuite 1 | Data segments rejected by receiver. |  |
| 28 | Auth2c | 10KB Red data, ciphersuite 1 both directions; bad receiver to sender key. | Like Auth1c but with ciphersuite 1 | Segments accepted by receiver; report segments rejected by the sender. |  |
| 29 | Auth3 | 10KB Red data, ciphersuite 255 in both directions; good keys. | Like Auth1 but with ciphersuite 255 | Ensure that authentication is invoked and that the block is received correctly. |  |
| 30 | RLEXC\_S | Block reception of Red data at receiver and let transmitter keep transmitting checkpoints until it gives up. | Block reception at receiver; wait for sender to give up and cancel the session after (re)transmitting a bunch of checkpoints; receiver never gets anything. |  |  |
| 31 | RLEXC\_R | Block reception of Red data after transmission of report but before report-acknowledgement; wait for receiver to give up after retransmitting a bunch of reports. | Drop the link in at least the forward direction after the first report but before the report acknowledgement segment ; provide no link queue to either the sender or receiver (elicit retransmissions of reports (from receiver) segments; cause retransmission limit exceeded RLEXC) | Report segments are retransmitted by the sender until the retransmission limit is exceeded. When the retransmission limit is exceeded the session is terminated and cancellation is sent. |  |
| 32 | RXMTCYCEXC\_S | Retransmission cyclesexceeded at sender (rexmitting checkpoints). | Turn the error rate up, RedPart size up, and segment size down and then jack with the MAXRXMTCYC value for the transmitter (set it significantly below that of the receiver) |  | Value of ION LTP RXMTCYC value, and its relationship to RLEXC, not clear. |
| 33 | RXMTCYCEXC\_R | Retransmission cycles exceeded at receiver (retransmitting reports) | Turn the error rate up, RedPart size up, and segment size down and then jack with the MAXRXMTCYC value for the receiver (set way below that of the transmitter |  | Value of ION LTP RXMTCYC value, and its relationship to RLEXC, not clear. |

# 

# Issues Identified During Testing and Implementation Notes

## Sender Response to RS from Closed Sessions

Once an LTP sender receives enough reception claims to cover the red-part of a block and has transmitted all of the block data at least once, it will close the transmit session. This means that the sender will no longer have a record of the destination LTP Engine ID for that session and, as a result, can not respond to future Report Segments from the receiver. This is an issue if the RA for the last RS is lost. The receiver will continue to retransmit RS until the retransmission limit is exceeded and the session is cancelled.

An approach that would require modification of the specification would be to include the receiving LTP engine ID in (at least some) segments sent by the LTP receiver to the LTP sender. If, for example, report segments contained the LTP Engine ID of the receiver, then the sender could respond to them even if the sending session had been closed. In this case there would still be the issue of HOW to respond, especially if the memory of closed session described above was not also implemented (e.g. a new cancel session reason – ‘No xmit session’?).

The current behavior is correct according to the specification. An implementation optimization to improve efficiency and eliminate the retransmitted Report Segments would be to keep, at the LTP sender, information about the mapping between session IDs and destination LTP engine IDs for some number of closed sessions. In this way, RA segments could be sent in response to retransmitted RS.

## Offsets of ReceptionClaims

The ION LTP implementation used ‘0’ as the offset for all reception claims, even if those claims appeared in a reception report with a non-zero lower bound. This was incorrect behavior relative to the specification and was fixed via a patch.

## HMAC/SHA1 Authentication

The HMAC/SHA1 authentication requires calculating the message authentication code from the SHA1 hash. Simply calculating the SHA1 hash is not enough. This is probably clear to people who deal with secure message signatures often but confused the developer of the Python implementation. The Python implementation was updated to generate the MAC from the hash.

## Authentication Trailer Length

The authentication trailer does NOT include the ciphersuite ID; it ONLY includes the actual authentication material. The length of the trailer is the length of the authentication material and does not include the extension type or length itself. Thus the length field of an authentication trailer for the HMAC-SHA1 ciphersuite should be 10 bytes. This is consistent with the text in section 3.1.4 of RFC5326:

Each extension consists of a one-octet tag identifying the type of the extension, followed by a length parameter in SDNV form, followed by a value of the specified length.

The Python implementation was updated to use the correct length fields in treailers.

## Mixed-Color Reception in ION

The initial ION implementation would not receive Green Data after it had received a report-ack for the red-part of the block. For example, for the transfer below, the ION receiver only admits to having received the first Green segment (packet capture at receiver).

LTP SID(18, 88) 0x0 Red data, NOT (Checkpoint, EORP or EOB) Data(0, 1186)

LTP SID(18, 88) 0x2 Red data, Checkpoint, EORP, NOT EOB Data(1186, 862) CP(8634)

LTP SID(18, 88) 0x8 Report segment RS(8634, 5427)

LTP SID(18, 88) x04 Green data, NOT EOB Data(2048, 1185)

LTP SID(18, 88) 0x9 Report-acknowledgment segment RA(5427)

LTP SID(18, 88) x04 Green data, NOT EOB Data(3233, 1185)

LTP SID(18, 88) x04 Green data, NOT EOB Data(4418, 1185)

LTP SID(18, 88) 0x7 Green data, EOB Data(5603, 541)

This was a bug; Scott provided a patch.

## Mixed-Color Transmission in ION

The initial ION sending implementation would ‘give up’ transmitting the GreenPart of a block once it received a report for the red part of the segment (packet capture at sender). In the example below, there was 4096 bytes of Green Data to be sent.

LTP SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) Data(0, 1192)

LTP SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB Data(1192, 856) CP(3792)

LTP SID(16, 2) x04 Green data, NOT EOB Data(2048, 1191)

LTP SID(16, 2) 0x8 Report segment RS(3792, 4077)

LTP SID(16, 2) 0x9 Report-acknowledgment segment RA(4077)

This was a bug; Scott provided a patch.

## Receiving Red Segment in Green Part of Block

ION seems willing to ignore a red segment received in the Green Part of a block. Keith thinks this should elicit a cancel from the receiver with reason MISCOLORED because the session ends at the receipt of the EOB.

LTPv0L SID(18, 14) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

LTPv0L SID(18, 14) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

LTPv0L SID(18, 14) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

LTPv0L SID(18, 14) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

LTPv0L SID(18, 14) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(10146)

LTPv0L SID(18, 14) 0x8 Report segment RS(10146, 8097) [0 , 5000]

LTPv0L SID(18, 14) x04 **Green data**, NOT EOB C(6) Data(**5000**, 1185)

LTPv0L SID(18, 14) 0x9 Report-acknowledgment segment RA(8097)

LTPv0L SID(18, 14) 0x0 **Red data**, NOT (Checkpoint, EORP or EOB) C(6) Data(**5000**, 1185)

LTPv0L SID(18, 14) x04 Green data, NOT EOB C(6) Data(6185, 1185)

LTPv0L SID(18, 14) x04 Green data, NOT EOB C(6) Data(7370, 1185)

LTPv0L SID(18, 14) x04 Green data, NOT EOB C(6) Data(8555, 1185)

LTPv0L SID(18, 14) 0x7 Green data, EOB C(6) Data(9740, 260)

Section 6.21of [1] states that the receiver SHOULD [not MUST] enqueue a cancellation segment with reason code MISCOLORED in response to the miscolored data. Therefore this behavior is compliant.

Later versions of ION issue a cancellation segment with reason code MISCOLORED.

## Green Segment in Red Space

The ION LTP implementation disregards rogue Green segments in the Red space if they are received after the report-ack for the legitimate red-part of the block is processed. Scott argues that this is within the letter of the spec (section 6.21 says that the receiver SHOULD [not MUST] enqueue a cancellation segment with reason code MISCOLORED in response to the miscolored segment).

The Python LTP receiver issues a CANCEL segment with reason code MISCOLORED because the green segment is in the red space.

In addition, ION delivered the correct red-part, then delivered the miscolored Green data to the application with the offset in the red space of the block.

Later versions of ION correct this problem. It was not tested here because the ION to Python test is enough to cover the interoperability requirement.

## Service Data Aggregation

The ION LTP implementation was prepending the SDNV encoding of the ‘real’ client service ID to **each** of the client PDUs in the aggregated block instead of including it only once at the beginning of the block.

The Red-3 version of the specification was updated to reflect the ION behavior (one client service ID per data capsule) in SDA, so that SDA can then handle ‘mixed’ collections of client data and hand them to (and extract them from) LTP. The Python implementation was updated to match the revised specification.

## Report Serial Number checks

The ION implementation does not cancel the session when the report serial number exceeds 232.

LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

LTPv0L SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(1192, 856) CP(2044)

LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(2048, 1191)

LTPv0L SID(16, 2) 0x8 Report segment RS(2044, 4294967302L) [0 , 2048]

LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(6)

/ LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(3239, 1191)

LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(4430, 1191)

LTPv0L SID(16, 2) 0x8 Report segment RS(2044, 4294967302L) [0 , 2048]

LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(6)

LTPv0L SID(16, 2) 0x7 Green data, EOB C(6) Data(5621, 523)

# Conclusions

By executing the tests described in Table 4-1, each of the interoperability requirements in Table 3-1 was verified in at least one direction, with the exception of those items listed in Table 3-2.

Detailed results of each of the tests are presented in Annex C.

1. Support File APIs
   1. Overview

This section contains an overview of the application interfaces for the various

* 1. pyltpsend

The pyltpsend application invokes the python LTP implementation to send blocks.

usage: pyltpsend.py [-h] [-a CHECKPOINTAMOUNT] [-e DESTENGINEID]

[--maxAggregationTime MAXAGGREGATIONTIME]

[--maxAggregationLength MAXAGGREGATIONLENGTH]

[-x MAXRXMTCYCSEND] [-L CPRXMTLIMIT] [-E MYENGINEID]

[-c DESTCLIENTID] [-C USERCANCELAFTER] [-n ENGINEMAP]

[-m ALLOWMIXEDCOLORS] [-g GREENPART] [-r REDPART]

[-R RATE] [-B TOKENBUCKETSIZE] [-s SEGMENTSIZE]

[-t SUSPENDTRANSMISSION SUSPENDTRANSMISSION SUSPENDTRANSMISSION]

[-A LOCALADDRESS] [-P LOCALPORT] [-H HEXTS] [-T TEXTS]

[-l LOGCONFIGURE LOGCONFIGURE]

Python-Based LTP Transmitter.

optional arguments:

-h, --help show this help message and exit

-a CHECKPOINTAMOUNT, --checkpointEvery CHECKPOINTAMOUNT

-e DESTENGINEID, --destEngineID DESTENGINEID

--maxAggregationTime MAXAGGREGATIONTIME

--maxAggregationLength MAXAGGREGATIONLENGTH

-x MAXRXMTCYCSEND, --maxRxmtCycSend MAXRXMTCYCSEND

-L CPRXMTLIMIT, --cpRxmtLimit CPRXMTLIMIT

-E MYENGINEID, --myEngineID MYENGINEID

-c DESTCLIENTID, --destClientID DESTCLIENTID

-C USERCANCELAFTER, --userCancelAfter USERCANCELAFTER

-n ENGINEMAP, --engineMap ENGINEMAP

-m ALLOWMIXEDCOLORS, --allowMixedColors ALLOWMIXEDCOLORS

-g GREENPART, --greenPart GREENPART

-r REDPART, --redPart REDPART

-R RATE, --rate RATE

-B TOKENBUCKETSIZE, --tokenBucketSize TOKENBUCKETSIZE

-s SEGMENTSIZE, --segmentSize SEGMENTSIZE

-t SUSPENDTRANSMISSION SUSPENDTRANSMISSION SUSPENDTRANSMISSION, --suspendTransmission SUSPENDTRANSMISSION SUSPENDTRANSMISSION SUSPENDTRANSMISSION

-A LOCALADDRESS, --localAddress LOCALADDRESS

-P LOCALPORT, --localPort LOCALPORT

-H HEXTS, --headerExts HEXTS

-T TEXTS, --trailerExts TEXTS

-l LOGCONFIGURE LOGCONFIGURE, --logConfigure LOGCONFIGURE LOGCONFIGURE

* 1. pyltprecv

The pyltprecv application invokes the python LTP implementation to receive blocks.

usage: pyltprecv.py [-h] [-e DESTENGINEID] [-E MYENGINEID]

[-c DESTCLIENTSERVICEID] [-C USERCANCELAFTER]

[-x MAXRXMTCYCRECV] [-n ENGINEMAP] [-S ENABLESDA]

[-t SUSPENDTRANSMISSION SUSPENDTRANSMISSION SUSPENDTRANSMISSION]

[-A LOCALADDRESS] [-P LOCALPORT] [-l LOGCONFIGURE]

Python-Based LTP Receiver.

optional arguments:

-h, --help show this help message and exit

-e DESTENGINEID, --destEngineID DESTENGINEID

-E MYENGINEID, --myEngineID MYENGINEID

-c DESTCLIENTSERVICEID, --destClientServiceID DESTCLIENTSERVICEID

-C USERCANCELAFTER, --userCancelAfter USERCANCELAFTER

-x MAXRXMTCYCRECV, --maxRxmtCycRecv MAXRXMTCYCRECV

-n ENGINEMAP, --engineMap ENGINEMAP

-S ENABLESDA, --enableSDA ENABLESDA

-t SUSPENDTRANSMISSION SUSPENDTRANSMISSION SUSPENDTRANSMISSION, --suspendTransmission SUSPENDTRANSMISSION SUSPENDTRANSMISSION SUSPENDTRANSMISSION

engineID, after, howLong

-A LOCALADDRESS, --localAddress LOCALADDRESS

-P LOCALPORT, --localPort LOCALPORT

-l LOGCONFIGURE, --logConfigure LOGCONFIGURE

* 1. ltpStart

The ltpstart program starts the ION LTP (engine) implementation.

PROGNAME [-i ionrcfile] [-l ltprcfile] [-s ionSecrcFile] [-h]

-i Use ion configuration file 'ionrcfile' (default=node16.ionrc).

-l Use ltp configuration file 'ltprcfile' (default=node16.ltprc).

-s Use ionsec security file 'ionSecrcfile'(default+).

-h Print this help.

* 1. ltpsend

The ltpsend application sends data via the ION LTP implementation.

ltpsend -d destEngineID [-c destClientService] -s stuffToSend [-r RED\_SIZE]

If stuffToSend begins with 'file:' then send file.

If not, make temp file and send string.

* 1. ltprecv

The ltprecv application receives data via the ION LTP implementation.

ltprecv -s clientServiceID [-d]

* 1. startIONLTPSender

The startIONLTPSender application starts the ION LTP engine implementation and an LTP sender to send data.

startIONLTPSender [-S ionSecFile] [-h]

-s Use ionsec security file 'ionSecFile' with LTP (authentication).

-h Print this help.

* 1. startIONSDASender

The startIONSDASender application starts the ION LTP engine implementation and an LTP sender to send data. The send operation uses the LTP Service Data Aggregation (SDA) mechanism.

startIONSDASender [-S ionSecFile] [-h]

-s Use ionsec security file 'ionSecFile' with LTP (authentication).

-h Print this help.

* 1. StartIONLTPReceiver

The startIONLTPReceiver application starts an ION LTP receiver with specific configuration

startIONLTPReceiver [-s clientServiceID] [ltpStartOptions] [-h]

-h Print this help.

* 1. startIONSDAReceiver

The startIONSDAReceiver application starts an ION LTP receiver with specific configuration, and using the LTP Service Data Aggregation (SDA) mechanism.

PROGNAME [-s clientServiceID] [sdaStartOptions] [-h]

-h Print this help.

* 1. udp\_encap\_over-udp\_gateway

The udp\_encap-over-udp\_gateway takes the user data portion of UDP packets and encapsulates them in CCSDS Encapsulation packets over UDP.

usage: udp\_encap-over-udp\_gateway.py [-h] [-L LOCAL\_LAN\_ADDR]

[--local-lan-port LOCAL\_LAN\_PORT]

[-l REMOTE\_LAN\_ADDR]

[--remote-lan-port REMOTE\_LAN\_PORT]

[-W LOCAL\_WAN\_ADDR]

[--local-wan-port LOCAL\_WAN\_PORT]

[-w REMOTE\_WAN\_ADDR]

[--remote-wan-port REMOTE\_WAN\_PORT]

[-p PROTOCOL]

Gateway

optional arguments:

-h, --help show this help message and exit

-L LOCAL\_LAN\_ADDR, --local-lan-addr LOCAL\_LAN\_ADDR

--local-lan-port LOCAL\_LAN\_PORT

-l REMOTE\_LAN\_ADDR, --remote-lan-addr REMOTE\_LAN\_ADDR

--remote-lan-port REMOTE\_LAN\_PORT

-W LOCAL\_WAN\_ADDR, --local-wan-addr LOCAL\_WAN\_ADDR

--local-wan-port LOCAL\_WAN\_PORT

-w REMOTE\_WAN\_ADDR, --remote-wan-addr REMOTE\_WAN\_ADDR

--remote-wan-port REMOTE\_WAN\_PORT

-p PROTOCOL, --protocol PROTOCOL

* 1. Gateway\_II

The gateway\_II application is a second implementation of the UDP to Encapsulation / Space Packets over UDP gateway.

INPUT FORMAT: -ll <listening lan IP> -lw <listening wan ip> -sw <sending/remote wan ip> -sl <remote lan ip> -ptype <EncapPacket/SpacePacket>

* 1. Ebtables\_block\_interfaces

The ebtables\_block\_interfaces program runs in parallel with a test and issues ebtables commands in response to timing or LTP segments seen.

usage: ebtables\_block\_interfaces.py [-h] [-d DPORT] [-S SKIP] [-n NUMTIMES]

[-i INTERFACE] [-I IN\_INTERFACE]

[-O OUT\_INTERFACE] [-s IP\_SRC]

[-D IP\_DST]

[-T TRIGGER] [-R RELEASE] [-t TIME] [-v]

Issue ebtables command in response to LTP packets seen.

optional arguments:

-h, --help show this help message and exit

-d DPORT, --dport DPORT

-S SKIP, --skip SKIP How many LTP TRIGGER segments to skip before starting

to block.

-n NUMTIMES, --numtimes NUMTIMES

How many times the particular 'trigger' can fire.

-i INTERFACE, --interface INTERFACE

Interface to watch for LTP segments.

-I IN\_INTERFACE, --in-interface IN\_INTERFACE

Interface to block (inbound).

-O OUT\_INTERFACE, --out-interface OUT\_INTERFACE

Interface to block (outbound).

-s IP\_SRC, --ip-src IP\_SRC

Src IP addr to block.

-D IP\_DST, --ip-dst IP\_DST

Dest IP addr to block.

-T TRIGGER, --trigger TRIGGER

Integer identifier of LTP trigger segment.

-R RELEASE, --release RELEASE

Integer identifier of LTP release (unblock) segment.

-t TIME, --time TIME How long to block the interface[s].

-v, --verbose Verbosity; multiple instances increases level.

* 1. injectLTPTraffic.py

The injectLTPTraffic executable runs in parallel with a test and injects replicated or modified LTP segments into the data stream.

usage: injectLTPTraffic.py [-h] [-S SKIP] [-f FUNCTION] [-s SEGMENTTYPE]

[-m MUNGEOFFSET] [-d DELAY] [-n NUMTIMES]

[-i INTERFACE] [-T TRIGGER] [-t EMBARGOTIME]

Inject (possibly mangled) LTP segment in response to segments seen.

optional arguments:

-h, --help show this help message and exit

-S SKIP, --skip SKIP

-f FUNCTION, --function [corrupt|miscolored] FUNCTION

-s SEGMENTTYPE, --segmentType SEGMENTTYPE

-m MUNGEOFFSET, --mungeOffset MUNGEOFFSET

-d DELAY, --delay DELAY

delay between trigger and effect

-n NUMTIMES, --numtimes NUMTIMES

-i INTERFACE, --interface INTERFACE

Interface to watch for LTP packets.

-T TRIGGER, --trigger TRIGGER

Integer identifier of LTP trigger segment.

-t EMBARGOTIME, --embargoTime EMBARGOTIME

How long to block after action.

usage: sendLinkCue.py [-h] [-I IPADDR] [-E ENGINEID] [-D DIRECTION] [-d DELAY]

[-e ENABLE]

* 1. sendLinkCue

The sendLinkCue application sends link cues to the Python LTP implementation.

Send Link Cues to the Python LTP implementation

optional arguments:

-h, --help show this help message and exit

-I IPADDR, --ipAddr IPADDR

-E ENGINEID, --engineID ENGINEID

-D DIRECTION, --direction DIRECTION

Direction: XMIT|RECV.

-d DELAY, --delay DELAY

delay before invoking.

-e ENABLE, --enable ENABLE

enable ON|OFF.

1. Configuration Files

This annex contains the configuration files used in the tests.

* 1. ionrc files
     1. Node16.ionrc

1 16

#1 16 ionconfig

s

m production 10

m consumption 10

@ +0

a range +0 +10000 18 16 2

a range +0 +10000 16 18 2

a range +0 +10000 16 19 2

a range +0 +10000 19 16 2

a contact +0 +1000 18 16 10000

a contact +0 +1000 16 18 10000

a contact +0 +1000 19 16 10000

a contact +0 +1000 16 19 10000

* + 1. node16\_basic1b.ionrc

1 16

#1 16 ionconfig

s

m production 10

m consumption 10

@ +0

a range +0 +10000 18 16 2

a range +0 +10000 16 18 2

a range +0 +10000 16 19 2

a range +0 +10000 19 16 2

# Connectivity from 0--10s

a contact +0 +10 18 16 10000

a contact +0 +10 16 18 10000

a contact +0 +10 19 16 10000

a contact +0 +10 16 19 10000

# No connectivity from 10--30s

# Connectivity from 30--1000s

a contact +30 +1000 18 16 10000

a contact +30 +1000 16 18 10000

a contact +30 +1000 19 16 10000

a contact +30 +1000 16 19 10000

* + 1. node16\_ntrSender.ionrc

1 16

#1 16 ionconfig

s

m production 10

m consumption 10

@ +0

a range +0 +10000 18 16 2

a range +0 +10000 16 18 2

a contact +0 +1000 16 18 10000

* 1. LTPRC files
     1. node16.ltprc

# 1

# est max export sessions

# database bytes needed

1 4 210000

# a span peer\_engine\_nbr max\_export\_sessions max\_export\_session\_block\_size max\_import\_sessions max\_import\_session\_block\_size max\_segment\_size aggregation\_size\_limit aggregation\_time\_limit 'LSO\_command' [queuing\_latency]

m screening n

# For talking directly (no encap gateway) to another engine ID (19)

# The '10000' in the udplso command is rate control.

a span 19 2 101000 2 101000 1200 1 1 'udplso 10.0.0.1:1113 10000'

# For use with the udp-to-encap-over-udp gateway

# The '10000' in the udplso command is rate control.

a span 18 2 101000 2 101000 1200 1 1 'udplso 127.0.0.1:2113 10000'

# ONE UDP input, on port INARDDR\_ANY:1113

s 'udplsi 0.0.0.0:1113'

# Turn on 'watch characters'

w 1

* + 1. node16.ltprc\_aggregation

# 1

# est max export sessions

# database bytes needed

1 4 210000

# a span peer\_engine\_nbr max\_export\_sessions max\_export\_session\_block\_size max\_import\_sessions max\_import\_session\_block\_size max\_segment\_size aggregation\_size\_limit aggregation\_time\_limit 'LSO\_command' [queuing\_latency]

m screening n

# For talking directly (no encap gateway) to another engine ID (19)

# The '10000' in the udplso command is rate control.

a span 19 2 101000 2 101000 1200 1 1 'udplso 10.0.0.1:1113 10000'

# For use with the udp-to-encap-over-udp gateway

# The '10000' in the udplso command is rate control.

# Aggregation size: 10000; aggregation time: 10s

#a span 18 2 101000 2 101000 1200 [Aggregation Size:] 100 [Aggregation Time:] 20 'udplso 127.0.0.1:2113 10000'

a span 18 2 101000 2 101000 1200 100 20 'udplso 127.0.0.1:2113 10000'

# ONE UDP input, on port INARDDR\_ANY:1113

s 'udplsi 0.0.0.0:1113'

# Turn on 'watch characters'

w 1

* + 1. node16.ltprc\_aggregation\_large

# 1

# est max export sessions

# database bytes needed

1 4 210000

# a span peer\_engine\_nbr max\_export\_sessions max\_export\_session\_block\_size max\_import\_sessions max\_import\_session\_block\_size max\_segment\_size aggregation\_size\_limit aggregation\_time\_limit 'LSO\_command' [queuing\_latency]

m screening n

# For talking directly (no encap gateway) to another engine ID (19)

# The '10000' in the udplso command is rate control.

a span 19 2 101000 2 101000 1200 1 1 'udplso 10.0.0.1:1113 10000'

# For use with the udp-to-encap-over-udp gateway

# The '10000' in the udplso command is rate control.

# Aggregation size: 10,000; aggregation time: 60s

#a span 18 2 101000 2 101000 1200 [Aggregation Size:] 100 [Aggregation Time:] 20 'udplso 127.0.0.1:2113 [RateControl]'

a span 18 2 101000 2 101000 1200 10000 60 'udplso 127.0.0.1:2113 1400'

# ONE UDP input, on port INARDDR\_ANY:1113

s 'udplsi 0.0.0.0:1113'

# Turn on 'watch characters'

w 1

* + 1. node16.ltprc\_aggregation\_time

# 1

# est max export sessions

# database bytes needed

1 4 210000

# a span peer\_engine\_nbr max\_export\_sessions max\_export\_session\_block\_size max\_import\_sessions max\_import\_session\_block\_size max\_segment\_size aggregation\_size\_limit aggregation\_time\_limit 'LSO\_command' [queuing\_latency]

m screening n

# For talking directly (no encap gateway) to another engine ID (19)

# The '10000' in the udplso command is rate control.

a span 19 2 101000 2 101000 1200 1 1 'udplso 10.0.0.1:1113 10000'

# For use with the udp-to-encap-over-udp gateway

# The '10000' in the udplso command is rate control.

# Aggregation size: 10000; aggregation time: 10s

#a span 18 2 101000 2 101000 1200 [Aggregation Size:] 100 [Aggregation Time:] 20 'udplso 127.0.0.1:2113 10000'

a span 18 2 101000 2 101000 1200 1000 20 'udplso 127.0.0.1:2113 10000'

# ONE UDP input, on port INARDDR\_ANY:1113

s 'udplsi 0.0.0.0:1113'

# Turn on 'watch characters'

w 1

* + 1. node16.ltprc\_slow

# 1

# est max export sessions

# database bytes needed

1 4 210000

# a span peer\_engine\_nbr max\_export\_sessions max\_export\_session\_block\_size max\_import\_sessions max\_import\_session\_block\_size max\_segment\_size aggregation\_size\_limit aggregation\_time\_limit 'LSO\_command' [queuing\_latency]

m screening n

# For talking directly (no encap gateway) to another engine ID (19)

# The '2000' in the udplso command is rate control.

a span 19 2 101000 2 101000 1200 1 1 'udplso 10.0.0.1:1114 3000'

# For use with the udp-to-encap-over-udp gateway

# The '2000' in the udplso command is rate control.

a span 18 2 101000 2 101000 1200 1 1 'udplso 127.0.0.1:2113 3000'

# ONE UDP input, on port INARDDR\_ANY:1113

s 'udplsi 0.0.0.0:1113'

# Turn on 'watch characters'

w 1

* 1. ionsec files
     1. node16.ionsecrc\_empty

# An empty ionsecrc database

1

* + 1. node16.ionsec

1

e 1

# add transmission rules

# a ltpxmitauthrule ltp\_engine\_id { ciphersuite frequency | ciphersuite key\_name frequency }

a ltpxmitauthrule 16 0 11 0

a ltpxmitauthrule 18 0 22 0

# add receiving rules

# a ltprecvauthrule ltp\_engine\_id { ciphersuite frequency | ciphersuite key\_name frequency }

a ltprecvauthrule 16 0 11 0

a ltprecvauthrule 18 0 11 0

#list all transmission rules

l ltpxmitauthrule

q

* + 1. node16.ionsecrc\_0\_11\_22

1

e 1

# add transmission rules

# a ltpxmitauthrule ltp\_engine\_id { ciphersuite frequency | ciphersuite key\_name frequency }

a ltpxmitauthrule 16 0 11 0

a ltpxmitauthrule 18 0 22 0

# add receiving rules

# a ltprecvauthrule ltp\_engine\_id { ciphersuite frequency | ciphersuite key\_name frequency }

a ltprecvauthrule 16 0 11 0

a ltprecvauthrule 18 0 11 0

#list all transmission rules

l ltpxmitauthrule

q

* + 1. node16.ionsecrc\_1\_tkey1\_rkey2

1

e 1

a key key1 key1\_pub.der

a key key2 key2\_pub.der

a key key3 key1\_priv.der

a key key4 key2\_priv.der

# add transmission rules

# a ltpxmitauthrule ltp\_engine\_id { ciphersuite frequency | ciphersuite key\_name frequency }

# If I don't send to myself, I only need s rule for transmitting DATA to other engines;

# in this case we'll use engine 18.

a ltpxmitauthrule 18 1 key3 0

# add receiving rules

# a ltprecvauthrule ltp\_engine\_id { ciphersuite frequency | ciphersuite key\_name frequency }

# The 16 rule is for when I'm sending, and receiving Reports from someone else;

# everybody I send to has to use the same key to send back to me

# The 18 rule is for when I'm receiving DATA segments from engine 18

a ltprecvauthrule 16 1 key2 0

a ltprecvauthrule 18 1 key1 0

#list all rules

l ltpxmitauthrule

l ltprecvauthrule

q

1. Detailed Test Results

# Test: Basic1

Send 100B block containing only Red (reliable) data.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Sender sends a 100-Byte Red LTP Block to a listening service on an LTP receiver. Exercises both implementations of the UDP-to-EncapOverUDP gateway to provide an interoperability test of the LTP over UDP LSA (pcaps over loopback) and LTP over Encapsulation LSA (pcaps between engines).  
  
For the Python->ION direction, send 'bogus' header / trailer extensions to ensure that the ION implementation ignores them.

## Test-Specific Success Criteria

For the Python -> ION run, ensure that there are bogus header / trailer extensions present. We could augment the injectLTPTraffic helper program if we needed to test this for both directions.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/gateway\_II -ll 127.0.0.1:2113 -lw 10.0.0.2:5555 -sw 10.0.0.1:5555 -sl 127.0.0.1:1113 -ptype EncapPacket &

sleep 5

./startIONLTPSender -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 773 |
| InitialReportSerialNumber | 3246 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(773, 0)] |
| allReports | [(3246, 773)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(773, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 3246) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [3246] |
| reports | [(3246, 773)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

efgshg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 773 |
| InitialReportSerialNumber | 3246 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(773, 0)] |
| allReports | [(3246, 773)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(773, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 3246) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [3246] |
| reports | [(3246, 773)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-28 09:53:21,707 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-28 09:53:33,182 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-28 09:53:33,183 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(100) EOB(1) md5(7acedd1a84a4cfcb6e7a16003242945e)

### Sender PCAP

09:53:32.643825 Ether / IP / UDP 10.0.0.2:58534 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(773)

09:53:33.696516 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(773, 3246) [0 , 100]

09:53:33.698136 Ether / IP / UDP 10.0.0.2:58534 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(3246)

### Sender PCAP over lo

09:53:32.643587 Ether / IP / UDP 127.0.0.1:39895 > 127.0.0.1:hsl\_storm /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(773)

09:53:33.696737 Ether / IP / UDP 127.0.0.1:34252 > 127.0.0.1:ltp\_deepspace /   
LTPv0L SID(16, 1) 0x8 Report segment RS(773, 3246) [0 , 100]

09:53:33.698076 Ether / IP / UDP 127.0.0.1:39895 > 127.0.0.1:hsl\_storm /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(3246)

### Receiver PCAP

09:53:33.143892 Ether / IP / UDP 10.0.0.2:58534 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(773)

09:53:33.196408 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(773, 3246) [0 , 100]

09:53:34.198215 Ether / IP / UDP 10.0.0.2:58534 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(3246)

### Receiver PCAP over lo

09:53:33.145195 Ether / IP / UDP 127.0.0.1:hsl\_storm > 127.0.0.1:ltp\_deepspace /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(773)

09:53:33.195859 Ether / IP / UDP 127.0.0.1:ltp\_deepspace > 127.0.0.1:hsl\_storm /   
LTPv0L SID(16, 1) 0x8 Report segment RS(773, 3246) [0 , 100]

09:53:34.200383 Ether / IP / UDP 127.0.0.1:hsl\_storm > 127.0.0.1:ltp\_deepspace /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(3246)

## Python to ION Results:

### Sender Commands

#!/bin/bash

echo "iptables rules for bar container" > bar\_iptables.txt

/usr/sbin/iptables -n -v --list >> bar\_iptables.txt

echo "end of iptables rules for bar container" >> bar\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

--bogusHeaderTrailer True \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 1

sync

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -n -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" >> foo\_iptables.txt

Python\_Files/supportFiles/gateway\_II -ll 127.0.0.1:2113 -lw 10.0.0.1:5555 -sw 10.0.0.2:5555 -sl 127.0.0.1:1113 -ptype EncapPacket &

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1504 |
| InitialReportSerialNumber | 4573 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1504, 0)] |
| allReports | [(4573, 1504)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(1504, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [12] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 4573) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [4573] |
| reports | [(4573, 1504)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 20 |
| sessionOriginator | 18 |
| trailerExtensions | [11] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414777084.429087 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 20)

\*\*\* 1414777084.429335 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414777084.431199 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414777084.637480 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 20)

\*\*\* 1414777085.650166 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 20)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1504 |
| InitialReportSerialNumber | 4573 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1504, 0)] |
| allReports | [(4573, 1504)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(1504, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [12] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 4573) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [4573] |
| reports | [(4573, 1504)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 20 |
| sessionOriginator | 18 |
| trailerExtensions | [11] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

stgs

### Sender PCAP

13:38:04.636943 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=12, L=3) Data(0, 100) CP(1504) TExt1 (T=11, L=8)

13:38:05.638567 Ether / IP / UDP 10.0.0.1:37044 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x8 Report segment RS(1504, 4573) [0 , 100]

13:38:05.649518 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x9 Report-acknowledgment segment RA(4573)

### Sender PCAP over lo

13:38:04.634937 Ether / IP / UDP 127.0.0.1:ltp\_deepspace > 127.0.0.1:hsl\_storm /   
LTPv0L SID(18, 20) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=12, L=3) Data(0, 100) CP(1504) TExt1 (T=11, L=8)

13:38:05.639715 Ether / IP / UDP 127.0.0.1:hsl\_storm > 127.0.0.1:ltp\_deepspace /   
LTPv0L SID(18, 20) 0x8 Report segment RS(1504, 4573) [0 , 100]

13:38:05.648144 Ether / IP / UDP 127.0.0.1:ltp\_deepspace > 127.0.0.1:hsl\_storm /   
LTPv0L SID(18, 20) 0x9 Report-acknowledgment segment RA(4573)

### Receiver PCAP

13:38:05.137005 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=12, L=3) Data(0, 100) CP(1504) TExt1 (T=11, L=8)

13:38:05.138516 Ether / IP / UDP 10.0.0.1:37044 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x8 Report segment RS(1504, 4573) [0 , 100]

13:38:06.149571 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x9 Report-acknowledgment segment RA(4573)

### Receiver PCAP over lo

13:38:05.137239 Ether / IP / UDP 127.0.0.1:34988 > 127.0.0.1:ltp\_deepspace /   
LTPv0L SID(18, 20) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=12, L=3) Data(0, 100) CP(1504) TExt1 (T=11, L=8)

13:38:05.138405 Ether / IP / UDP 127.0.0.1:34207 > 127.0.0.1:hsl\_storm /   
LTPv0L SID(18, 20) 0x8 Report segment RS(1504, 4573) [0 , 100]

13:38:06.149736 Ether / IP / UDP 127.0.0.1:34988 > 127.0.0.1:ltp\_deepspace /   
LTPv0L SID(18, 20) 0x9 Report-acknowledgment segment RA(4573)

## Comments:

### General Comments

Note the loopback packet captures; these show LTP over UDP encapsulation.  
  
The implementations both RECEIVE on UDP port 1113; they transmit to a different port on the gateway because the python gateway binds to INARDDR\_ANY. This could be fixed by having the gateways bind to 127.0.0.2 and listen on port 1113, and having the LTP implementations transmit to 127.0.0.2.

### Python to ION Comments

ION LTP doesn't issue an explicit ReceptionSessionStart.indication  
  
The Python to ION test inserts 'bogus' extensions to test that the ION implementation silently ignores them.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) | Ensure: Segment Types: 3, 8, 9 | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | Good: Header / Trailer Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  | Ensure that for Python -> ION, bogus header and trailer extensions are present. | No Extensions | CHECKME -- header / trailer extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  | Ensure: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  | Ensure: RSN RSN\_CSN | Good: RSN RSN\_CSN [(3246, 773)] | Good: RSN RSN\_CSN [(4573, 1504)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  | Ensure: Report Acknowledgement(s) sent | Good: Report Acknowledgement(s) sent: [3246] | Good: Report Acknowledgement(s) sent: [4573] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | Good: trailer format for unknown extension |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  | Ensure: TransmissionRequest.request | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation.request primitive issued. | No cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  | Ensure: StartTransmission | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  | Ensure: Checkpoint transmitted | Good: Checkpoint transmitted: [(773, 0)] | Good: Checkpoint transmitted: [(1504, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  | Ensure: Report transmitted | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  | Ensure: SendReceptionReport | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  | Ensure: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  | Ensure: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  | Ensure: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  | Ensure: ReceptionSessionStart.indication | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  | Note the loopback packet captures; these show LTP over UDP encapsulation. | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  | The implementations both RECEIVE on UDP port 1113; they transmit to a different port on the gateway. | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  | Ensure: SessionNumber OK | Good: SessionNumber OK (1) | Good: SessionNumber OK (20) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  | Ensure: InitialCheckpointSerialNumber OK | Good: InitialCheckpointSerialNumber OK (773) | Good: InitialCheckpointSerialNumber OK (1504) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  | Ensure: look for randomness across all runs. | Good: InitialCheckpointSerialNumber: 773 | Good: InitialCheckpointSerialNumber: 1504 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  | Ensure unknown extensions in Python -> ION run. | No UNKNOWN extensions found | Good: -- unknown extension(s) [11, 12] found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  | Ensure: No CCSDS Space Packet | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  | Ensure: CCSDS ENCAP | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Basic1B - Encap

Send a 100B block containing only Red (reliable) with link cues. Block the link from [0, 45]; suspend transmission at sender during [10, 30]. Use LTP over Encapsulation packets and two different encapsulation implementations.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Use ebtables to block the link from sender to receiver from t=0 to t=45s.  
  
In the LTP implementation, suspend transmission between t=10 and t=30 seconds [configurable in Python; use spans in ion].  
  
Use the two different implementations of LTP over Space Packet Encapsulation to exercise the Space Packet LSA.

## Test-Specific Success Criteria

Because of the link cues, there should be no retransmissions during the time [10, 30].  
Once the link is restored, data transfer should resume and complete successfully.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -i node16\_basic1b.ionrc -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/gateway\_II -ll 127.0.0.1:2113 -lw 10.0.0.1:5555 -sw 10.0.0.2:5555 -sl 127.0.0.1:1113 -ptype EncapPacket >& gateway\_II.out &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

./Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -s 10.0.0.2 -S 0 -t 45

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2543 |
| InitialReportSerialNumber | 15217 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2543, 0), (2543, 0), (2543, 0)] |
| allReports | [(15217, 2543)] |
| allSegmentsInOrder | [3L, 3L, 3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(2543, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 15217) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [15217] |
| reports | [(15217, 2543)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 3, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16\_basic1b.ionrc

ltpstart using ION rc file:, node16\_basic1b.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

efg=g=gshg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2543 |
| InitialReportSerialNumber | 15217 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2543, 0)] |
| allReports | [(15217, 2543)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(2543, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 15217) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [15217] |
| reports | [(15217, 2543)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-30 18:23:24,573 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-30 18:24:02,054 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-30 18:24:02,055 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(100) EOB(1) md5(7acedd1a84a4cfcb6e7a16003242945e)

### Sender PCAP

18:23:31.506347 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(2543)

18:23:36.503346 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(2543)

18:24:01.508631 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(2543)

18:24:02.565586 Ether / IP / UDP 10.0.0.1:43600 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(2543, 15217) [0 , 100]

18:24:02.567848 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(15217)

### Receiver PCAP

18:24:02.008703 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(2543)

18:24:02.065524 Ether / IP / UDP 10.0.0.1:43600 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(2543, 15217) [0 , 100]

18:24:03.067899 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(15217)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encap &

tcpdump -i lo -w python\_lo.pcap &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

--suspendTransmission 16 10 20 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/gateway\_II -ll 127.0.0.1:2113 -lw 10.0.0.1:5555 -sw 10.0.0.2:5555 -sl 127.0.0.1:1113 -ptype EncapPacket >& gateway\_II.out &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

./Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -s 10.0.0.1 -s 10.0.0.2 -S 0 -t 45

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 5853 |
| InitialReportSerialNumber | 661 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(5853, 0), (5853, 0), (5853, 0), (5853, 0), (5853, 0), (5853, 0), (5853, 0)] |
| allReports | [(661, 5853)] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(5853, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 661) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [661] |
| reports | [(661, 5853)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 7, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 61 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414707199.768425 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 61)

\*\*\* 1414707199.768608 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414707199.769986 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414707199.770361 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=0.000420)

\*\*\* 1414707199.979835 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 61)

\*\*\* 1414707200.771784 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=1.001804)

\*\*\* 1414707201.772811 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=2.002870)

\*\*\* 1414707202.774081 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=3.004090)

\*\*\* 1414707203.774881 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=4.004924)

\*\*\* 1414707204.776548 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=5.006590)

\*\*\* 1414707205.778229 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=6.008276)

\*\*\* 1414707206.779855 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=7.009878)

\*\*\* 1414707207.781475 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=8.011518)

\*\*\* 1414707208.783138 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=9.013166)

\*\*\* 1414707209.784828 - pyltpsend - INFO - Suspend xmission to 16 after 10.000000 (cur=10.014873)

\*\*\* 1414707233.046159 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 61)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 5853 |
| InitialReportSerialNumber | 661 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(5853, 0)] |
| allReports | [(661, 5853)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(5853, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 661) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [661] |
| reports | [(661, 5853)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 61 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

stgs

### Sender PCAP

18:13:19.976075 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(5853)

18:13:21.977644 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(5853)

18:13:23.982633 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(5853)

18:13:25.985006 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(5853)

18:13:27.989397 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(5853)

18:13:50.011892 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(5853)

18:13:52.014866 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(5853)

18:13:53.017518 Ether / IP / UDP 10.0.0.1:40078 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x8 Report segment RS(5853, 661) [0 , 100]

18:13:53.046869 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x9 Report-acknowledgment segment RA(661)

### Receiver PCAP

18:13:52.514931 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(5853)

18:13:52.517385 Ether / IP / UDP 10.0.0.1:40078 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x8 Report segment RS(5853, 661) [0 , 100]

18:13:53.546931 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 61) 0x9 Report-acknowledgment segment RA(661)

## Comments:

### Python to ION Comments

Link cue timings hand-tuned due to the delays in test execution.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(15217, 2543)] | Good: RSN RSN\_CSN [(661, 5853)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [15217] | Good: Report Acknowledgement(s) sent: [661] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation.request primitive issued. | No cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(2543, 0), (2543, 0), (2543, 0)] | Good: Checkpoint transmitted: [(5853, 0), (5853, 0), (5853, 0), (5853, 0), (5853, 0), (5853, 0), (5853, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  | Ensure: verify via pcap inspection. | Good: Sender stops transmitting Red Segments | Good: Sender stops transmitting Red Segments |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  | Ensure: verify via pcap inspection. | Good: Sender suspends Checkpoint timer. | Good: Sender suspends Checkpoint timer. |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  | Ensure: verify via pcap inspection. | Good: Sender resumes checkpoint timer. | Good: Sender resumes checkpoint timer. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  | Ensure: checkpoint Retransmission | Good: checkpoints retransmitted: [(2543, 3, 15217)] | Good: checkpoints retransmitted: [(5853, 7, 661)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (61) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (2543) | Good: InitialCheckpointSerialNumber OK (5853) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 2543 | Good: InitialCheckpointSerialNumber: 5853 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  | Ensure: No CCSDS Space Packet | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  | Ensure: No CCSDS Space Packet | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  | Ensure: CCSDS Space Packet | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Basic1B - Space

Send a 100B block containing only Red (reliable) with link cues. Block the link from [0, 45]; suspend transmission at sender during [10, 30]. Use LTP over Space Packet Encapsulation and two different encapsulation implementations.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Use ebtables to block the link from sender to receiver from t=0 to t=45s.  
  
In LTP implementation, suspend transmission between t=10 and t=30 seconds [configurable in Python; use spans in ion].  
  
Use the two different implementations of LTP over Space Packet Encapsulation to exercise the Space Packet LSA.

## Test-Specific Success Criteria

Because of the link cues, there should be no retransmissions during the time [10, 30].  
Once the link is restored, data transfer should resume and complete successfully.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/gateway\_II -ll 127.0.0.1:2113 -lw 10.0.0.2:5556 -sw 10.0.0.1:5556 -sl 127.0.0.1:1113 -ptype SpacePacket &

sleep 1

./startIONLTPSender -i node16\_basic1b.ionrc -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5556 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5556 \

--protocol=spacePacket &

sleep 1

tcpdump -i lo -w python\_lo.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

./Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 --ip-dst 10.0.0.1 -S 0 -t 45

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 7474 |
| InitialReportSerialNumber | 10312 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(7474, 0), (7474, 0), (7474, 0)] |
| allReports | [(10312, 7474)] |
| allSegmentsInOrder | [3L, 3L, 3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(7474, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | SPACE\_PACKET |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 10312) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [10312] |
| reports | [(10312, 7474)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 3, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16\_basic1b.ionrc

ltpstart using ION rc file:, node16\_basic1b.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

efg=g=gshg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 7474 |
| InitialReportSerialNumber | 10312 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(7474, 0)] |
| allReports | [(10312, 7474)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(7474, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | SPACE\_PACKET |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 10312) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [10312] |
| reports | [(10312, 7474)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-30 18:25:41,517 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-30 18:26:23,997 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-10-30 18:26:23,999 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 2) len(100) EOB(1) md5(7acedd1a84a4cfcb6e7a16003242945e)

### Sender PCAP

18:25:53.459307 Ether / IP / UDP 10.0.0.2:36343 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(7474)

18:25:58.456470 Ether / IP / UDP 10.0.0.2:36343 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(7474)

18:26:23.462343 Ether / IP / UDP 10.0.0.2:36343 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(7474)

18:26:24.509191 Ether / IP / UDP 10.0.0.1:5556 > 10.0.0.2:5556 / CCSDSSPACE /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7474, 10312) [0 , 100]

18:26:24.511177 Ether / IP / UDP 10.0.0.2:36343 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(10312)

### Receiver PCAP

18:26:23.962412 Ether / IP / UDP 10.0.0.2:36343 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(7474)

18:26:24.009120 Ether / IP / UDP 10.0.0.1:5556 > 10.0.0.2:5556 / CCSDSSPACE /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7474, 10312) [0 , 100]

18:26:25.011223 Ether / IP / UDP 10.0.0.2:36343 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(10312)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5556 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5556 \

--protocol=spacePacket &

tcpdump -i lo -w python\_lo.pcap &

sleep 10

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

--suspendTransmission 16 15 25 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/gateway\_II -ll 127.0.0.1:2113 -lw 10.0.0.1:5556 -sw 10.0.0.2:5556 -sl 127.0.0.1:1113 -ptype SpacePacket >& gateway\_II.out &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

./Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -s 10.0.0.2 -S 0 -t 45

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 12315 |
| InitialReportSerialNumber | 7031 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(12315, 0), (12315, 0), (12315, 0), (12315, 0), (12315, 0), (12315, 0), (12315, 0), (12315, 0), (12315, 0)] |
| allReports | [(7031, 12315), (7031, 12315)] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 8L, 9L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(12315, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | SPACE\_PACKET |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 7031) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [7031] |
| reports | [(7031, 12315)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 9, 4: 0, 5: 0, 6: 0, 7: 0, 8: 2, 9: 2, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 9 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414708086.116801 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 9)

\*\*\* 1414708086.117008 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414708086.118205 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414708086.118581 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=0.000420)

\*\*\* 1414708086.325027 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 9)

\*\*\* 1414708087.120110 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=1.001905)

\*\*\* 1414708088.121502 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=2.003323)

\*\*\* 1414708089.123343 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=3.005137)

\*\*\* 1414708090.124730 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=4.006552)

\*\*\* 1414708091.126684 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=5.008504)

\*\*\* 1414708092.128602 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=6.010424)

\*\*\* 1414708093.130604 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=7.012404)

\*\*\* 1414708094.132261 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=8.014081)

\*\*\* 1414708095.134224 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=9.016021)

\*\*\* 1414708096.136205 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=10.018020)

\*\*\* 1414708097.138202 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=11.019997)

\*\*\* 1414708098.140213 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=12.022025)

\*\*\* 1414708099.142201 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=13.024002)

\*\*\* 1414708100.144238 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=14.026056)

\*\*\* 1414708101.146328 - pyltpsend - INFO - Suspend xmission to 16 after 15.000000 (cur=15.028094)

\*\*\* 1414708128.393485 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 9)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 12315 |
| InitialReportSerialNumber | 7031 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(12315, 0)] |
| allReports | [(7031, 12315), (7031, 12315)] |
| allSegmentsInOrder | [3L, 8L, 9L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(12315, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | SPACE\_PACKET |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 7031) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [7031] |
| reports | [(7031, 12315)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 2, 9: 2, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 9 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

stg+g+g

### Sender PCAP

18:28:06.322685 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:08.325563 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:10.327981 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:12.333040 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:14.335298 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:16.342108 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:18.344911 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:20.347071 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:47.364907 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:48.368535 Ether / IP / UDP 10.0.0.1:52049 > 10.0.0.2:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x8 Report segment RS(12315, 7031) [0 , 100]

18:28:48.391469 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(7031)

18:28:52.535991 Ether / IP / UDP 10.0.0.1:52049 > 10.0.0.2:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x8 Report segment RS(12315, 7031) [0 , 100]

18:28:52.563575 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(7031)

### Receiver PCAP

18:28:47.864965 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(0, 100) CP(12315)

18:28:47.868475 Ether / IP / UDP 10.0.0.1:52049 > 10.0.0.2:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x8 Report segment RS(12315, 7031) [0 , 100]

18:28:48.891531 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(7031)

18:28:52.035932 Ether / IP / UDP 10.0.0.1:52049 > 10.0.0.2:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x8 Report segment RS(12315, 7031) [0 , 100]

18:28:53.063646 Ether / IP / UDP 10.0.0.2:5556 > 10.0.0.1:5556 / CCSDSSPACE /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(7031)

## Comments:

### Python to ION Comments

Link cue timings hand-tuned due to the delays in test execution.  
  
UDP to Space Packet gateway issue prevented reception of report-ack, causing receiver to retransmit report.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(12198, 4400)] | Good: RSN RSN\_CSN [(8021, 8284), (8021, 8284)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [12198] | Good: Report Acknowledgement(s) sent: [8021] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation.request primitive issued. | No cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(4400, 0), (4400, 0), (4400, 0)] | Good: Checkpoint transmitted: [(8284, 0), (8284, 0), (8284, 0), (8284, 0), (8284, 0), (8284, 0), (8284, 0), (8284, 0), (8284, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | Good: StartReportSegment Timer (Report(s) retransmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  | Ensure: verify via pcap inspection. | Good: Sender stops transmitting Red Segments | Good: Sender stops transmitting Red Segments |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  | Ensure: verify via pcap inspection. | Good: Sender suspends Checkpoint timer. | Good: Sender suspends Checkpoint timer. |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  | Ensure: verify via pcap inspection. | Good: Sender resumes checkpoint timer. | Good: Sender resumes checkpoint timer. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  | Ensure: checkpoint Retransmission | Good: checkpoints retransmitted: [(4400, 3, 12198)] | Good: checkpoints retransmitted: [(8284, 9, 8021)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | Good: Report retransmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No cancel due to RLEXC |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | No check: error in receiving gateway prevents Report ACK reception. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (58) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (4400) | Good: InitialCheckpointSerialNumber OK (8284) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 4400 | Good: InitialCheckpointSerialNumber: 8284 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  | Ensure: CCSDS Space Packet | Good: CCSDS Space Packet | Good: CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS APID assignment yet | No CCSDS APID assignment yet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  | Ensure: No CCSDS Encap | No CCSDS Encap | No CCSDS Encap |

# Test: Basic2

Send a 1024B block containing only Green (unreliable) data

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 0 | (Bytes) |
| GREEN\_SIZE | 1024 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Sender sends a 1024 byte LTP Block containing only Green (unreliable) data.

## Test-Specific Success Criteria

Correct receipt of the green data.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -s file:redAndGreenParts -r 0

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(0, 1023)] |
| AllRedDataSegments | [] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 1023 |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [7L] |
| cancelReasonCodes | [] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | False |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 1023 |
| highestRedByteIndex | False |
| lastCancelOrAck | None |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 1, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [7L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

efg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(0, 1023)] |
| AllRedDataSegments | [] |
| AmountGreenDataReceivedByApp | 1024 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 1023 |
| EORP | None |
| GreenSegmentsReceived | [(0, 1023)] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [7L] |
| cancelReasonCodes | [] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | False |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 1023 |
| highestRedByteIndex | False |
| lastCancelOrAck | None |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 1, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [7L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-28 09:42:58,125 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-28 09:43:05,598 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-28 09:43:05,599 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1024) EOB(1) md5(b2ea9f7fcea831a4a63b213f41a8855b)

\*\*\* 2014-10-28 09:43:05,599 - pyltprecv - INFO - Green Data Extents received for session (16, 1): (0, 1024)

### Sender PCAP

09:43:05.063396 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x7 Green data, EOB C(6) Data(0, 1024)

### Receiver PCAP

09:43:05.563467 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x7 Green data, EOB C(6) Data(0, 1024)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-g file:greenPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(0, 1023)] |
| AllRedDataSegments | [] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 1023 |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [7L] |
| cancelReasonCodes | [] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | False |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 1023 |
| highestRedByteIndex | False |
| lastCancelOrAck | None |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 1, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [7L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 89 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414504307.582943 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 89)

\*\*\* 1414504307.583306 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414504307.585391 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414504307.587175 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 89)

\*\*\* 1414504308.694442 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 89)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(0, 1023)] |
| AllRedDataSegments | [] |
| AmountGreenDataReceivedByApp | 1024 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 1023 |
| EORP | None |
| GreenSegmentsReceived | [(0, 1023)] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [7L] |
| cancelReasonCodes | [] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | False |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 1023 |
| highestRedByteIndex | False |
| lastCancelOrAck | None |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 1, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [7L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 89 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

s

### Sender PCAP

09:51:48.693696 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 89) 0x7 Green data, EOB C(6) Data(0, 1024)

### Receiver PCAP

09:51:49.193769 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 89) 0x7 Green data, EOB C(6) Data(0, 1024)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 7 | Good: Segment Types: 7 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 7 | Good: Segment Types: 7 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length Data | Good: CSID Offset Length Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No Reports [] | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation.request primitive issued. | No cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | No checkpoints transmitted | No checkpoints transmitted |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No checkpoints transmitted | No checkpoints transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted . | No checkpoints retransmitted . |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  | Ensure: GreenPartReception.indication | Good: GreenPartReception.indication | Good: GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report sent. | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (89) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | No checkpoints | No checkpoints |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | No checkpoints | No checkpoints |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Basic3

Block with 2kBRed and 4kBGreen

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 2048 | (Bytes) |
| GREEN\_SIZE | 4096 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Sender sends an LTP Block containing a mix of Red and Green data.

## Test-Specific Success Criteria

Demonstrates use of 1- and 2-byte SDNVs in red-part segments.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -s file:redAndGreenParts -r 2048

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(2048, 3238), (3239, 4429), (4430, 5620), (5621, 6143)] |
| AllRedDataSegments | [(0, 1191), (1192, 2047)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 6143 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3955 |
| InitialReportSerialNumber | 2743 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'GreenPartSegmentArrival', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(3955, 0)] |
| allReports | [(2743, 3955)] |
| allSegmentsInOrder | [0L, 2L, 4L, 8L, 9L, 4L, 4L, 7L] |
| cancelReasonCodes | [] |
| checkpoints | [(3955, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 6143 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 2743) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [2743] |
| reports | [(2743, 3955)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 1, 3: 0, 4: 3, 5: 0, 6: 0, 7: 1, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeefgggshgggg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(2048, 3238), (3239, 4429), (4430, 5620), (5621, 6143)] |
| AllRedDataSegments | [(0, 1191), (1192, 2047)] |
| AmountGreenDataReceivedByApp | 4096 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 6143 |
| EORP | 2047 |
| GreenSegmentsReceived | [(2048, 3238), (3239, 4429), (4430, 5620), (5621, 6143)] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3955 |
| InitialReportSerialNumber | 2743 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'GreenPartSegmentArrival', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(3955, 0)] |
| allReports | [(2743, 3955)] |
| allSegmentsInOrder | [0L, 2L, 8L, 4L, 9L, 4L, 4L, 7L] |
| cancelReasonCodes | [] |
| checkpoints | [(3955, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 6143 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 2743) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [2743] |
| reports | [(2743, 3955)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 1, 3: 0, 4: 3, 5: 0, 6: 0, 7: 1, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-28 09:59:36,377 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-28 09:59:43,850 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-28 09:59:44,818 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(2048) EOB(0) md5(1576a94d6cb334dd126cb1c27f19e0f2)

\*\*\* 2014-10-28 09:59:45,537 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1191) EOB(0) md5(df2d55384ca1bcb5362bd5480cbb708d)

\*\*\* 2014-10-28 09:59:46,523 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1191) EOB(0) md5(6b98158cad4c21e5af65e568b0fd432e)

\*\*\* 2014-10-28 09:59:47,540 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1191) EOB(0) md5(98fe31cdecc45f0d822e632198c97702)

\*\*\* 2014-10-28 09:59:48,508 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(523) EOB(1) md5(242fc03ba937e956d6d8de40e7eb07ac)

\*\*\* 2014-10-28 09:59:48,508 - pyltprecv - INFO - Green Data Extents received for session (16, 1): (2048, 4096)

### Sender PCAP

09:59:43.302697 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

09:59:44.285556 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(1192, 856) CP(3955)

09:59:45.002671 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(2048, 1191)

09:59:45.322928 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(3955, 2743) [0 , 2048]

09:59:45.985567 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(2743)

09:59:46.013092 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(3239, 1191)

09:59:46.995840 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(4430, 1191)

09:59:47.979545 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x7 Green data, EOB C(6) Data(5621, 523)

### Receiver PCAP

09:59:43.802738 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

09:59:44.785598 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(1192, 856) CP(3955)

09:59:44.822890 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(3955, 2743) [0 , 2048]

09:59:45.502733 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(2048, 1191)

09:59:46.485619 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(2743)

09:59:46.513176 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(3239, 1191)

09:59:47.495883 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(4430, 1191)

09:59:48.479595 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x7 Green data, EOB C(6) Data(5621, 523)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-g file:greenPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(2048, 3232), (3233, 4417), (4418, 5602), (5603, 6143)] |
| AllRedDataSegments | [(0, 1185), (1186, 2047)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 6143 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4817 |
| InitialReportSerialNumber | 876 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4817, 0)] |
| allReports | [(876, 4817)] |
| allSegmentsInOrder | [0L, 2L, 8L, 4L, 9L, 4L, 4L, 7L] |
| cancelReasonCodes | [] |
| checkpoints | [(4817, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 6143 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 876) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [876] |
| reports | [(876, 4817)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 1, 3: 0, 4: 3, 5: 0, 6: 0, 7: 1, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 82 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1415030994.453811 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 82)

\*\*\* 1415030994.454109 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1415030994.456455 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1415030997.621599 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 82)

\*\*\* 1415031000.798304 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 82)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(2048, 3232), (3233, 4417), (4418, 5602), (5603, 6143)] |
| AllRedDataSegments | [(0, 1185), (1186, 2047)] |
| AmountGreenDataReceivedByApp | 4096 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 6143 |
| EORP | 2047 |
| GreenSegmentsReceived | [(2048, 3232), (3233, 4417), (4418, 5602), (5603, 6143)] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4817 |
| InitialReportSerialNumber | 876 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4817, 0)] |
| allReports | [(876, 4817)] |
| allSegmentsInOrder | [0L, 2L, 8L, 4L, 9L, 4L, 4L, 7L] |
| cancelReasonCodes | [] |
| checkpoints | [(4817, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 6143 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 876) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [876] |
| reports | [(876, 4817)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 1, 3: 0, 4: 3, 5: 0, 6: 0, 7: 1, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 82 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sstgsssss

### Sender PCAP

11:09:55.666630 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

11:09:56.573246 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(1186, 862) CP(4817)

11:09:57.575928 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x8 Report segment RS(4817, 876) [0 , 2048]

11:09:57.781199 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) x04 Green data, NOT EOB C(6) Data(2048, 1185)

11:09:57.781481 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x9 Report-acknowledgment segment RA(876)

11:09:58.987782 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) x04 Green data, NOT EOB C(6) Data(3233, 1185)

11:10:00.193299 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) x04 Green data, NOT EOB C(6) Data(4418, 1185)

11:10:00.796788 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x7 Green data, EOB C(6) Data(5603, 541)

### Receiver PCAP

11:09:56.166697 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

11:09:57.073296 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(1186, 862) CP(4817)

11:09:57.075881 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x8 Report segment RS(4817, 876) [0 , 2048]

11:09:58.281259 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) x04 Green data, NOT EOB C(6) Data(2048, 1185)

11:09:58.281505 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x9 Report-acknowledgment segment RA(876)

11:09:59.487827 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) x04 Green data, NOT EOB C(6) Data(3233, 1185)

11:10:00.693361 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) x04 Green data, NOT EOB C(6) Data(4418, 1185)

11:10:01.296822 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 82) 0x7 Green data, EOB C(6) Data(5603, 541)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 2 , 4 , 7 , 8 , 9 | Good: Segment Types: 0 , 2 , 4 , 7 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 2 , 4 , 7 , 8 , 9 | Good: Segment Types: 0 , 2 , 4 , 7 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(2743, 3955)] | Good: RSN RSN\_CSN [(876, 4817)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [2743] | Good: Report Acknowledgement(s) sent: [876] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation.request primitive issued. | No cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(3955, 0)] | Good: Checkpoint transmitted: [(4817, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | Good: GreenPartReception.indication | Good: GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (82) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (3955) | Good: InitialCheckpointSerialNumber OK (4817) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 3955 | Good: InitialCheckpointSerialNumber: 4817 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Basic4

Multi-checkpoint Red data

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 2048 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Send an LTP block containing Red data wehre a checkpoint is generated DURING the red-part of the data transmission, and not only at the end.

## Test-Specific Success Criteria

Ensure that the sender generates a "Checkpoint, not EORP, not EOB" segment (type 1) and that the receiver responds to it correctly.

## There Are No ION to Python Results:

There are no results for this test.

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

--checkpointEvery 1000\

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2047)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2305 |
| InitialReportSerialNumber | 12865 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2305, 0), (2306, 0)] |
| allReports | [(12865, 2305), (12866, 2306)] |
| allSegmentsInOrder | [1L, 3L, 8L, 9L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(2305, 0), (2306, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 12866) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [12865, 12866] |
| reports | [(12865, 2305), (12866, 2306)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 1, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 2, 9: 2, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [1L, 3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 39 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414076505.299195 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 39)

\*\*\* 1414076505.299458 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414076505.301023 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414076507.417296 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 39)

\*\*\* 1414076508.440113 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 39)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2047)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2305 |
| InitialReportSerialNumber | 12865 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2305, 0), (2306, 0)] |
| allReports | [(12865, 2305), (12866, 2306)] |
| allSegmentsInOrder | [1L, 8L, 3L, 8L, 9L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(2305, 0), (2306, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 12866) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [12865, 12866] |
| reports | [(12865, 2305), (12866, 2306)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 1, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 2, 9: 2, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [1L, 3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 39 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

Running ionadmin with cmdFileName: node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

logSize: 0

logKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sgstgss

### Sender PCAP

11:01:46.509280 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(0, 1186) CP(2305)

11:01:47.418116 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(2306)

11:01:47.512877 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x8 Report segment RS(2305, 12865) [0 , 1186]

11:01:47.564925 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x9 Report-acknowledgment segment RA(12865)

11:01:48.421073 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x8 Report segment RS(2306, 12866) [0 , 2048]

11:01:48.440535 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x9 Report-acknowledgment segment RA(12866)

### Receiver PCAP

11:01:47.009360 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(0, 1186) CP(2305)

11:01:47.012810 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x8 Report segment RS(2305, 12865) [0 , 1186]

11:01:47.918181 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(2306)

11:01:47.921011 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x8 Report segment RS(2306, 12866) [0 , 2048]

11:01:48.064989 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x9 Report-acknowledgment segment RA(12865)

11:01:48.940618 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x9 Report-acknowledgment segment RA(12866)

## Comments:

### Python to ION Comments

ION LTP implementation does not support multi-checkpointing of red data.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  |  | Good: Segment Types: 1 , 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  |  | Good: Segment Types: 1 , 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  |  | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  |  | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  |  | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  |  | Good: RSN RSN\_CSN [(12865, 2305), (12866, 2306)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  |  | Good: Report Acknowledgement(s) sent: [12865, 12866] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  |  | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  |  | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  |  | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  |  | No cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  |  | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  |  | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  |  | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  |  | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  |  | Good: Checkpoint transmitted: [(2305, 0), (2306, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  |  | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  |  | Good: StartReportSegment Timer (Report(s) retransmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  |  | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  |  | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  |  | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  |  | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  |  | No cancel due to RLEXC |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: RedPartReception.indication |  | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  |  | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  | Ensure: SendReceptionReport |  | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  |  | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  |  | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  |  | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  |  | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  |  | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  |  | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  |  | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  |  | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  |  | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  |  | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  |  | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  |  | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  |  | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  |  | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  |  | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  |  | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication |  | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication |  | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  |  | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  |  | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  |  | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  |  | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  |  | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  |  | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  |  | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  |  | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  |  | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  |  | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  |  | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  |  | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  |  | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  |  | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  |  | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  |  | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  |  | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  |  | Good: SessionNumber OK (39) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  |  | Good: InitialCheckpointSerialNumber OK (2305) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  |  | Good: InitialCheckpointSerialNumber: 2305 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  |  | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  |  | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  |  | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  |  | Good: CCSDS ENCAP |

# Test: NTR

Red data to non-transmitting receiver. A single checkpoint segment should be generated and sent, then checkpoint retransmission immediately suspended.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 2048 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Sender is configured to know that receiver can't transmit when it sends the block.

## Test-Specific Success Criteria

Checkpoint timers at the sender should be immediately suspended (there should be asingle checkpoint transmission and no checkpoint retransmsissions).

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -i node16\_ntrSender.ionrc -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': False}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2047)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 6260 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [6260] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(6260, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 3L] |
| cancelReasonCodes | [] |
| checkpoints | [(6260, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16\_ntrSender.ionrc

ltpstart using ION rc file:, node16\_ntrSender.ionrc

Running ionadmin with cmdFileName: node16\_ntrSender.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

logSize: 0

logKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eefgg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2047)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 6260 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [6260] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(6260, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 3L] |
| cancelReasonCodes | [] |
| checkpoints | [(6260, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-23 11:04:02,717 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-23 11:04:11,198 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-23 11:04:12,168 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(2048) EOB(1) md5(1576a94d6cb334dd126cb1c27f19e0f2)

### Sender PCAP

11:04:10.661636 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

11:04:11.644478 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6260)

### Receiver PCAP

11:04:11.161711 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

11:04:12.144565 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6260)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': False, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -i node16\_18to16only.ionrc

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2047)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1755 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [1755] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1755, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 3L] |
| cancelReasonCodes | [] |
| checkpoints | [(1755, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 23 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414076867.067946 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 23)

\*\*\* 1414076867.068309 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414076867.069750 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414076869.184094 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 23)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2047)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1755 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [1755] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1755, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 3L] |
| cancelReasonCodes | [] |
| checkpoints | [(1755, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | None |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 23 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16\_18to16only.ionrc

ltpstart using ION rc file:, node16\_18to16only.ionrc

Running ionadmin with cmdFileName: node16\_18to16only.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

logSize: 0

logKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sst

### Sender PCAP

11:07:48.278237 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 23) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

11:07:49.184092 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 23) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(1755)

### Receiver PCAP

11:07:48.778310 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 23) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

11:07:49.684154 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 23) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(1755)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 3 | Good: Segment Types: 0 , 3 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 3 | Good: Segment Types: 0 , 3 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN Data | Good: CSID Offset Length CSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No Reports [] | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation.request primitive issued. | No cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(6260, 0)] | Good: Checkpoint transmitted: [(1755, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  | Exactly 1 checkpoint sent (receiver can't transmit reports) | Good: Verified by hand: only 1 checkpoint sent. | Good: Verified by hand: only 1 checkpoint sent. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report sent. | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (23) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (6260) | Good: InitialCheckpointSerialNumber OK (1755) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 6260 | Good: InitialCheckpointSerialNumber: 1755 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: NTS

Red data from non-transmitting sender (for the purposes of report ACK generation). A single report segment should be generated and sent, then report retransmission immediately suspended.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 2048 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Configure the receiver to know that the sender is not allowed to transmit (report acknowledgements).

## Test-Specific Success Criteria

Receiver will send ONE report before suspending report timer.  
  
Sender may retransmit checkpoint multiple times.  
  
Sender may cancel the session due to too many retransmissions.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -i node16.ionrc -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': False, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -T 8 -s 10.0.0.1 -t 500

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 6792 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [6792] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L] |
| cancelReasonCodes | [2, 2] |
| checkpoints | [(6792, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 2, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16.ionrc

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eefgg=g=g=g=g=g=g=g=g=g{ggg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047), (1192, 2047)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 6792 |
| InitialReportSerialNumber | 3169 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [3169] |
| allCheckpoints | [(6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0)] |
| allReports | [(3169, 6792)] |
| allSegmentsInOrder | [0L, 3L, 8L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 13L, 12L, 13L] |
| cancelReasonCodes | [2, 2] |
| checkpoints | [(6792, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | 13 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [(3169, 6792)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 0, 10: 0, 11: 0, 12: 2, 13: 2, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-24 08:37:11,446 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-24 08:37:19,918 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-10-24 08:37:20,887 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 2) len(2048) EOB(1) md5(1576a94d6cb334dd126cb1c27f19e0f2)

\*\*\* 2014-10-24 08:38:10,929 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-24 08:38:10,929 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 08:38:10,929 - pyltprecv - INFO - (0, 2048)

\*\*\* 2014-10-24 08:38:15,949 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-24 08:38:15,949 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 08:38:15,950 - pyltprecv - INFO -

\*\*\* 2014-10-24 08:38:20,920 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-24 08:38:20,920 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 08:38:20,920 - pyltprecv - INFO -

### Sender PCAP

08:37:19.390266 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

08:37:20.373311 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:25.385837 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:30.386738 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:35.387912 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:40.389069 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:45.390113 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:50.391118 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:55.394772 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:38:00.396977 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:38:05.398056 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:38:10.400811 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

08:38:15.402137 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

### Receiver PCAP

08:37:19.890368 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

08:37:20.873394 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:20.899620 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(6792, 3169) [0 , 2048]

08:37:25.885895 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:30.886825 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:35.887979 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:40.889120 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:45.890180 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:50.891153 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:37:55.894824 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:38:00.897028 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:38:05.898128 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1192, 856) CP(6792)

08:38:10.900887 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

08:38:10.933511 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender

08:38:15.902214 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

08:38:15.952061 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'OWLT': 1}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -i node16\_16to18only.ionrc

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -T 8 -s 10.0.0.1 -t 500

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 9103 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [9103] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L] |
| cancelReasonCodes | [2] |
| checkpoints | [(9103, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 72 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414154490.857439 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 72)

\*\*\* 1414154490.857572 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414154490.858396 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414154492.978298 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 72)

\*\*\* 1414154535.089944 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 72) reason(2)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047), (1186, 2047)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 2047 |
| EORP | 2047 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 9103 |
| InitialReportSerialNumber | 234 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [234] |
| allCheckpoints | [(9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0)] |
| allReports | [(234, 9103)] |
| allSegmentsInOrder | [0L, 3L, 8L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 13L] |
| cancelReasonCodes | [2] |
| checkpoints | [(9103, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2047 |
| highestRedByteIndex | 2047 |
| lastCancelOrAck | 13 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [(234, 9103)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 0, 10: 0, 11: 0, 12: 1, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 72 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16\_16to18only.ionrc

ltpstart using ION rc file:, node16\_16to18only.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sstgsssssssssssssssssssss}g

### Sender PCAP

08:41:32.069478 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

08:41:32.978315 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:34.985090 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:36.989548 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:38.994158 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:41.001245 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:43.008637 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:45.012180 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:47.020175 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:49.023077 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:51.026597 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:53.033695 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:55.040313 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:57.047207 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:59.049595 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:01.052932 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:03.059845 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:05.066979 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:07.069668 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:09.074197 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:11.078887 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:13.084640 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:15.095709 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0xC Cancel segment from block sender Reason(RLEXC)

### Receiver PCAP

08:41:32.569598 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

08:41:33.478390 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:33.481410 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x8 Report segment RS(9103, 234) [0 , 2048]

08:41:35.485153 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:37.489645 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:39.494225 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:41.501323 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:43.508698 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:45.512250 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:47.520260 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:49.523142 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:51.526657 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:53.533765 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:55.540378 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:57.547266 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:41:59.549666 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:01.552980 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:03.559916 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:05.567029 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:07.569722 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:09.574254 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:11.579010 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:13.584701 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(1186, 862) CP(9103)

08:42:15.595769 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0xC Cancel segment from block sender Reason(RLEXC)

08:42:15.598586 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 72) 0xD Cancel-acknowledgment segment to block sender

## Comments:

### ION to Python Comments

Test blocks transmission to sender to help verify that only 1 report is generated. Sender continues to retransmit checkpoint and eventually cancels session with RLEXC.

### Python to ION Comments

Test blocks transmission to sender to help verify that only 1 report is generated. Sender continues to retransmit checkpoint and eventually cancels session with RLEXC.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 3 , 12 | Good: Segment Types: 0 , 3 , 12 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 3 , 12 | Good: Segment Types: 0 , 3 , 12 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN [(3169, 6792)] | Good: RSN [(234, 9103)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | Good: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation.request primitive issued. | No cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0), (6792, 0)] | Good: Checkpoint transmitted: [(9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0), (9103, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | Good: checkpoints retransmitted. | Good: checkpoints retransmitted. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  | Verify that exactly 1 report is sent (sender can't ACK report) | Good: Verified by inspectionof pcap file. | Good: Verified by inspectionof pcap file. |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(6792, 10, False)] | Good: checkpoints retransmitted: [(9103, 21, False)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report ack | No report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check -- receiver has path to sender. | No check -- receiver has path to sender. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | Good: CancelACK to sender | Good: CancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No check: no cancel-ack received by sender due to blocked link. | No check: no cancel-ack received by sender due to blocked link. |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting after cancel. | Good: Sender stops transmitting after cancel. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (72) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (6792) | Good: InitialCheckpointSerialNumber OK (9103) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 6792 | Good: InitialCheckpointSerialNumber: 9103 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Cancel1

User cancel at sender during red-part transmission.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 10000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 500 | (Bytes/sec) |
| SEG\_SIZE | 400 | (Bytes) |

## Test Procedure

After starting the transfer and while it is ongoing, drop the link (without providing any link cues to the sender) and then cancel the session at the sender (by the sending LTP user). After a short period of time (to allow for retransmission of cancellation segments) restore the link.

## Test-Specific Success Criteria

Cancellation segments sent by sender; cancellation segments retransmitted by sender while link down; cancellation ACK sent by receiver after link is restored; sender ceases transmission of cancellation segments after receiving cancellation ACK. No data carrying segments sent by sender after user cancel processed; a single cancel-ACK sent in response to a given cancel segment. Cancellation indications delivered to sending and receiving LTP clients.

## There Are No ION to Python Results:

There are no results for this test.

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 500 -s 400 \

-r file:redPart \

--userCancelAfter 5 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -s 10.0.0.1 -s 10.0.0.2 --skip 5 -t 10

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 385), (386, 770), (771, 1155), (1156, 1540), (1541, 1925), (1926, 2310)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 13L] |
| cancelReasonCodes | [0, 0, 0, 0, 0, 0, 0] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 2310 |
| lastCancelOrAck | 13 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 6, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 7, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [0L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 98 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414082096.806797 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 98)

\*\*\* 1414082096.807001 - pyltpsend - INFO - Setting send rate control to 500.000000 bytes per second.

\*\*\* 1414082096.808675 - pyltpsend - INFO - Sender userCancelAfter is 5.0

\*\*\* 1414082096.808830 - pyltpsend - INFO - Checking for sender userCancelAfter (0.000160)

\*\*\* 1414082097.810089 - pyltpsend - INFO - Checking for sender userCancelAfter (1.001379)

\*\*\* 1414082098.811682 - pyltpsend - INFO - Checking for sender userCancelAfter (2.002993)

\*\*\* 1414082099.813060 - pyltpsend - INFO - Checking for sender userCancelAfter (3.004366)

\*\*\* 1414082100.814592 - pyltpsend - INFO - Checking for sender userCancelAfter (4.005894)

\*\*\* 1414082101.815839 - pyltpsend - INFO - Checking for sender userCancelAfter (5.007142)

\*\*\* 1414082101.816213 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 98) reason(0)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 385), (386, 770), (771, 1155), (1156, 1540), (1541, 1925)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 12L, 13L] |
| cancelReasonCodes | [0] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 1925 |
| lastCancelOrAck | 13 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 5, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [0L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 98 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

Running ionadmin with cmdFileName: node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

logSize: 0

logKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

ssssss}g

### Sender PCAP

12:34:57.618943 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 386)

12:34:58.426798 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(386, 385)

12:34:59.234415 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(771, 385)

12:35:00.041981 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1156, 385)

12:35:00.850128 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1541, 385)

12:35:01.660834 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1926, 385)

12:35:01.821333 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

12:35:03.828325 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

12:35:05.834141 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

12:35:07.838824 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

12:35:09.843781 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

12:35:11.846594 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

12:35:13.850567 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

12:35:14.854482 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xD Cancel-acknowledgment segment to block sender

### Receiver PCAP

12:34:58.119016 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 386)

12:34:58.926882 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(386, 385)

12:34:59.734535 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(771, 385)

12:35:00.542052 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1156, 385)

12:35:01.350179 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1541, 385)

12:35:14.350635 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

12:35:14.354423 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 98) 0xD Cancel-acknowledgment segment to block sender

## Comments:

### ION to Python Comments

ION LTP implementation does not support user cancel.

### Python to ION Comments

Note: link is dropped for a while to elicit retransmission of cancel segments by the sender.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  |  | Good: Segment Types: 0 , 12 , 13 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) | Segment Types: 0, 12, 13 |  | Good: Segment Types: 0 , 12 , 13 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  |  | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  |  | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  |  | Good: CSID Offset Length Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  |  | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  |  | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Cancel Reason Code 0x00 |  | Good: CancelReasonCodes: 0x00 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  |  | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  |  | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  | Ensure: Cancellation.request primitive issued |  | Good: Cancellation.request primitive issued. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  |  | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  | Verified by hand -- client service exists at receiver |  | No check -- cleint service exists at receiver. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  | Verified by hand -- client service exists at receiver |  | No check -- cleint service exists at receiver. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  |  | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  |  | No checkpoints transmitted |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  |  | No checkpoints transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  |  | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  |  | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  |  | No checkpoints retransmitted . |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  |  | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  |  | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  |  | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  |  | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  |  | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  |  | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  |  | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  |  | NO InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  |  | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  |  | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  |  | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  |  | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  |  | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  |  | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  |  | No check -- receiver has path to sender. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  | Ensure: CancelACK to sender |  | Good: CancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  |  | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  | Ensure: no cancel sent after cancel ack received. |  | Good: no cancel sent after cancel ack received. |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  | Ensure: no data after cancel sent. |  | Good: Sender stops transmitting after cancel. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  |  | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  |  | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  |  | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: No RedPartReception.indication |  | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: No TransmissionSessionCompletion.indication |  | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  | Ensure: TransmissionSessionCancellation.indication |  | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  | Ensure: ReceptionSessionCancellation.indication |  | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  |  | NO InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  |  | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  |  | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  |  | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  |  | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  |  | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  |  | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  |  | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  |  | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  |  | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  |  | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  |  | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  |  | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  |  | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  |  | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  |  | Good: SessionNumber OK (98) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  |  | No checkpoints |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  |  | No checkpoints |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  |  | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  |  | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  |  | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  |  | Good: CCSDS ENCAP |

# Test: Cancel2

User cancel at receiver during red-part data transmission; drop link to elicit retransmission of cancel segments.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 10000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 500 | (Bytes/sec) |
| SEG\_SIZE | 400 | (Bytes) |

## Test Procedure

After starting the transfer and while it is ongoing, drop the link (without providing any link cues to the sender) and then cancel the session at the receiver (by the receiving LTP user). After a short period of time (to allow for retransmission of cancellation segments) restore the link.

## Test-Specific Success Criteria

Ensure that cancellation segments are retransmitted by the receiver while the link is down; ensure that a cancellation ACK is sent by sender after the link is restored; ensure that the receiver ceases transmission of cancellation segments after receiving the cancellation ACK. No segments shold be sent by the receiver after the cancel ACK is received; ensure that a single cancel-ACK sent in response to a given cancel segment.  
  
Cancellation indications should be delivered to sending and receiving LTP clients.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -l node16.ltprc\_slow -s file:redAndGreenParts &

while true; do

ping -c 1 10.0.0.1

# sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -C 20 -n "{16: {'OWLT': 3, 'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -s 10.0.0.1 -S 2 -t 10

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 14L, 15L] |
| cancelReasonCodes | [0] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 7146 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 6, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: l

Using ltprc file node16.ltprc\_slow

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeefggggggs]g

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [0L, 0L] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 14L, 0L, 0L, 14L, 15L] |
| cancelReasonCodes | [0, 0] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 7146 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 6, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 2, 15: 1} |
| segmentTypes | [0L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-28 10:43:14,662 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-28 10:43:14,666 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 0.000003 [20.000000])

\*\*\* 2014-10-28 10:43:14,666 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:14,666 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:15,667 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 1.001507 [20.000000])

\*\*\* 2014-10-28 10:43:15,668 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:15,668 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:16,668 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 2.002505 [20.000000])

\*\*\* 2014-10-28 10:43:16,668 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:16,669 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:17,669 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 3.003507 [20.000000])

\*\*\* 2014-10-28 10:43:17,670 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:17,670 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:18,671 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 4.005010 [20.000000])

\*\*\* 2014-10-28 10:43:18,671 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:18,671 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:19,673 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 5.006773 [20.000000])

\*\*\* 2014-10-28 10:43:19,673 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:19,673 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:20,673 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 6.007638 [20.000000])

\*\*\* 2014-10-28 10:43:20,674 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:20,674 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:21,675 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 7.009434 [20.000000])

\*\*\* 2014-10-28 10:43:21,676 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:21,676 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:22,677 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(None -- 8.011294 [20.000000])

\*\*\* 2014-10-28 10:43:22,677 - pyltprecv - INFO - %%%%%%%%% No LTPSessionReceiver for client service ID: 6

\*\*\* 2014-10-28 10:43:22,678 - pyltprecv - INFO - %%%%%%%%% SDAEnabled is: False

\*\*\* 2014-10-28 10:43:23,138 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-28 10:43:23,678 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 9.012536 [20.000000])

\*\*\* 2014-10-28 10:43:24,680 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 10.013948 [20.000000])

\*\*\* 2014-10-28 10:43:25,681 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 11.015329 [20.000000])

\*\*\* 2014-10-28 10:43:26,683 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 12.016800 [20.000000])

\*\*\* 2014-10-28 10:43:27,684 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 13.018232 [20.000000])

\*\*\* 2014-10-28 10:43:28,686 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 14.019729 [20.000000])

\*\*\* 2014-10-28 10:43:29,687 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 15.021171 [20.000000])

\*\*\* 2014-10-28 10:43:30,689 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 16.022640 [20.000000])

\*\*\* 2014-10-28 10:43:31,690 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 17.024232 [20.000000])

\*\*\* 2014-10-28 10:43:32,692 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 18.025728 [20.000000])

\*\*\* 2014-10-28 10:43:33,693 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 19.027226 [20.000000])

\*\*\* 2014-10-28 10:43:34,695 - pyltprecv - INFO - %%%%%%%%%%% pyltprecv cancel loop @ top...(<ltp.LTPSessionReceiver instance at 0xb73c980c> -- 20.028756 [20.000000])

\*\*\* 2014-10-28 10:43:34,695 - pyltprecv - INFO - %%%%%%%%%%% Cancelling receive session (16, 1)

\*\*\* 2014-10-28 10:43:34,696 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 1) reason USR\_CNCLD

\*\*\* 2014-10-28 10:43:34,696 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-28 10:43:34,696 - pyltprecv - INFO - (0, 4765)

### Sender PCAP

10:43:22.610286 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

10:43:25.885487 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

10:43:29.159878 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

10:43:32.435112 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

10:43:35.710119 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

10:43:38.985316 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

10:43:41.203274 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(USR\_CNCLD)

10:43:42.260503 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xF Cancel-acknowledgment segment to block receiver

### Receiver PCAP

10:43:23.110352 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

10:43:26.385557 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

10:43:29.659951 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

10:43:32.935164 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

10:43:34.701903 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(USR\_CNCLD)

10:43:36.210186 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

10:43:39.485393 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

10:43:40.703226 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(USR\_CNCLD)

10:43:42.760574 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xF Cancel-acknowledgment segment to block receiver

## There Are No Python to ION Results:

There are no results for this test.

## Comments:

### ION to Python Comments

SDA disabled on receiving LTP engine to facilitate cancellation timer machinery.

### Python to ION Comments

ION LTP implementation does not support user cancel.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 14 , 15 |  |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 14 , 15 |  |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions |  |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions |  |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length Data |  |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No Reports [] |  |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements |  |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes : 0x00 | Good: CancelReasonCodes: 0x00 |  |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers |  |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request |  |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  | Ensure: Cancellation.request primitive issued | Good: verified by hand |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. |  |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission |  |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | No checkpoints transmitted |  |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No checkpoints transmitted |  |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports transmitted |  |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted |  |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted . |  |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. |  |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No reports transmitted |  |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted |  |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No RedPartReception.indication |  |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication |  |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent |  |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent |  |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication |  |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission |  |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report sent. |  |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions |  |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check -- receiver has path to sender. |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  | Ensure: CancelACK to receiver | Good: CancelACK to receiver |  |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | Good: no cancel sent after cancel ack received. |  |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  | Ensure: no data after cancel sent. | Good: Sender stops transmitting after cancel. |  |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession |  |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present |  |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication |  |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: No RedPartReception.indication | No RedPartReception.indication |  |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: No TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |  |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  | Ensure: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |  |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  | Ensure: ReceptionSessionCancellation.indication | Good: ReceptionSessionCancellation.indication |  |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication |  |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA |  |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram |  |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram |  |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA |  |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 |  |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth |  |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite |  |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check |  |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check |  |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check |  |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. |  |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure |  |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. |  |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception |  |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) |  |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | No checkpoints |  |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | No checkpoints |  |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found |  |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet |  |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet |  |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP |  |

# Test: Cancel3

User cancel at sender with a non-transmitting receiver during red-part data transmission. That is, the sender knows that the receiver is not allowed to transmit when the session is canceled.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 10000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 500 | (Bytes/sec) |
| SEG\_SIZE | 400 | (Bytes) |

## Test Procedure

After starting the transfer and while it is ongoing, cancel the session at the sender (by the sending LTP user). Receiver is non-transmitting and sender knows it.

## Test-Specific Success Criteria

A single cancel segment sent by sender. Cancellation indication delivered to sending LTP client.

## There Are No ION to Python Results:

There are no results for this test.

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 500 -s 400 \

-r file:redPart \

--userCancelAfter 5 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': False, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -i node16\_18to16only.ionrc

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 385), (386, 770), (771, 1155), (1156, 1540), (1541, 1925), (1926, 2310)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 12L] |
| cancelReasonCodes | [0] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 2310 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 6, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 47 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414088246.323185 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 47)

\*\*\* 1414088246.323324 - pyltpsend - INFO - Setting send rate control to 500.000000 bytes per second.

\*\*\* 1414088246.324347 - pyltpsend - INFO - Sender userCancelAfter is 5.0

\*\*\* 1414088246.325882 - pyltpsend - INFO - Checking for sender userCancelAfter (0.000417)

\*\*\* 1414088247.327174 - pyltpsend - INFO - Checking for sender userCancelAfter (1.002796)

\*\*\* 1414088248.328549 - pyltpsend - INFO - Checking for sender userCancelAfter (2.004187)

\*\*\* 1414088249.330889 - pyltpsend - INFO - Checking for sender userCancelAfter (3.006502)

\*\*\* 1414088250.332225 - pyltpsend - INFO - Checking for sender userCancelAfter (4.007865)

\*\*\* 1414088251.333474 - pyltpsend - INFO - Checking for sender userCancelAfter (5.009125)

\*\*\* 1414088251.333793 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 47) reason(0)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 385), (386, 770), (771, 1155), (1156, 1540), (1541, 1925), (1926, 2310)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 12L] |
| cancelReasonCodes | [0] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 2310 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 6, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 47 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16\_18to16only.ionrc

ltpstart using ION rc file:, node16\_18to16only.ionrc

Running ionadmin with cmdFileName: node16\_18to16only.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

logSize: 0

logKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sssssss}

### Sender PCAP

14:17:27.131434 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 386)

14:17:27.936603 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(386, 385)

14:17:28.742348 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(771, 385)

14:17:29.546665 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1156, 385)

14:17:30.351146 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1541, 385)

14:17:31.156631 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1926, 385)

14:17:31.340367 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

### Receiver PCAP

14:17:27.631485 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 386)

14:17:28.436741 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(386, 385)

14:17:29.242497 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(771, 385)

14:17:30.046736 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1156, 385)

14:17:30.851221 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1541, 385)

14:17:31.656714 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1926, 385)

14:17:31.840423 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

## Comments:

### ION to Python Comments

ION LTP implementation does not support user cancel.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  |  | Good: Segment Types: 0 , 12 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  |  | Good: Segment Types: 0 , 12 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  |  | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  |  | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  |  | Good: CSID Offset Length Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  |  | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  |  | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x00 |  | Good: CancelReasonCodes: 0x00 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  |  | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  |  | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  | Ensure: Cancellation.request primitive issued |  | Good: verified by hand |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  |  | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  |  | No check -- cleint service exists at receiver. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  |  | No check -- cleint service exists at receiver. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  |  | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  |  | No checkpoints transmitted |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  |  | No checkpoints transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  |  | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  |  | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  |  | No checkpoints retransmitted . |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  |  | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  |  | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  |  | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  |  | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  |  | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  |  | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  |  | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  |  | NO InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  |  | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  |  | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  |  | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  | Ensure: no cancel segment retransmission |  | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  |  | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  |  | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  |  | No check -- receiver has path to sender. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  |  | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  |  | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  |  | No cancel ack received (was looking for 13 in response to 12 in [0L, 0L, 0L, 0L, 0L, 0L, 12L]) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  | Ensure: no data after cancel sent. |  | Good: Sender stops transmitting after cancel. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  |  | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  |  | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  |  | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  |  | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  |  | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  |  | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  |  | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  |  | NO InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  |  | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  |  | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  |  | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  |  | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  |  | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  |  | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  |  | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  |  | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  |  | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  |  | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  |  | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  |  | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  |  | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  |  | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  |  | Good: SessionNumber OK (47) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  |  | No checkpoints |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  |  | No checkpoints |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  |  | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  |  | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  |  | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  |  | Good: CCSDS ENCAP |

# Test: Cancel4

User cancellation at sender before any data is sent (red-part data submitted).

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 10000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 500 | (Bytes/sec) |
| SEG\_SIZE | 400 | (Bytes) |

## Test Procedure

After sending the transmission request but before any data can be sent, sender cancels.

## Test-Specific Success Criteria

No LTP segments are ever transmitted.

## There Are No ION to Python Results:

There are no results for this test.

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 500 -s 400 \

-r file:redPart \

--userCancelAfter 5 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': False, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -i node16\_18to16only.ionrc

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 385), (386, 770), (771, 1155), (1156, 1540), (1541, 1925), (1926, 2310)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 12L] |
| cancelReasonCodes | [0] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 2310 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 6, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 47 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414088246.323185 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 47)

\*\*\* 1414088246.323324 - pyltpsend - INFO - Setting send rate control to 500.000000 bytes per second.

\*\*\* 1414088246.324347 - pyltpsend - INFO - Sender userCancelAfter is 5.0

\*\*\* 1414088246.325882 - pyltpsend - INFO - Checking for sender userCancelAfter (0.000417)

\*\*\* 1414088247.327174 - pyltpsend - INFO - Checking for sender userCancelAfter (1.002796)

\*\*\* 1414088248.328549 - pyltpsend - INFO - Checking for sender userCancelAfter (2.004187)

\*\*\* 1414088249.330889 - pyltpsend - INFO - Checking for sender userCancelAfter (3.006502)

\*\*\* 1414088250.332225 - pyltpsend - INFO - Checking for sender userCancelAfter (4.007865)

\*\*\* 1414088251.333474 - pyltpsend - INFO - Checking for sender userCancelAfter (5.009125)

\*\*\* 1414088251.333793 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 47) reason(0)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 385), (386, 770), (771, 1155), (1156, 1540), (1541, 1925), (1926, 2310)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 12L] |
| cancelReasonCodes | [0] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 2310 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 6, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 47 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16\_18to16only.ionrc

ltpstart using ION rc file:, node16\_18to16only.ionrc

Running ionadmin with cmdFileName: node16\_18to16only.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

logSize: 0

logKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sssssss}

### Sender PCAP

14:17:27.131434 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 386)

14:17:27.936603 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(386, 385)

14:17:28.742348 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(771, 385)

14:17:29.546665 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1156, 385)

14:17:30.351146 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1541, 385)

14:17:31.156631 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1926, 385)

14:17:31.340367 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

### Receiver PCAP

14:17:27.631485 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 386)

14:17:28.436741 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(386, 385)

14:17:29.242497 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(771, 385)

14:17:30.046736 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1156, 385)

14:17:30.851221 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1541, 385)

14:17:31.656714 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1926, 385)

14:17:31.840423 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 47) 0xC Cancel segment from block sender Reason(USR\_CNCLD)

## Comments:

### ION to Python Comments

ION LTP implementation does not support user cancel.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) | Verify: No Segments |  | No Segments |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  |  | No Segments |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  |  | Good: Verified by hand |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  | Ensure: Cancellation.request primitive issued |  | Good: verified by hand |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  |  | Good: verified by hand |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  |  | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  |  | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  |  | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  |  | No Segments |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  |  | No Segments |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No Segments |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  |  | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  |  | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  |  | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  |  | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  |  | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  |  | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  |  | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  |  | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  |  | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  |  | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  |  | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  |  | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  |  | No segments |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  |  | No segments |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  |  | No segments |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  |  | No segments |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  |  | No segments |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  |  | No segments |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  |  | No segments |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  |  | No segments |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  |  | No segments |

# Test: Error1

10KRed, 5KGreen, 20% packet loss rate.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 20 | (plr) |
| RED\_SIZE | 10000 | (Bytes) |
| GREEN\_SIZE | 5000 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

## Test-Specific Success Criteria

Correct reception of the Red-Part Data; receipt of at least some Green-Part Data.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -r 10000 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(10000, 11190), (11191, 12381), (12382, 13572), (13573, 14763), (14764, 14999)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146), (7147, 8337), (8338, 9528), (9529, 9999), (9529, 9999), (1192, 2382), (3574, 4764), (8338, 9527), (9528, 9528), (9528, 9528), (9528, 9528), (1192, 2381), (2382, 2382)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 14999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 7747 |
| InitialReportSerialNumber | 10927 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'GreenPartSegmentArrival', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(7747, 0), (7747, 0), (7748, 10927), (7748, 10927), (7748, 10927), (7749, 10928)] |
| allReports | [(10927, 7747), (10928, 7748), (10929, 7749)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 2L, 4L, 4L, 4L, 4L, 7L, 2L, 8L, 9L, 0L, 0L, 0L, 1L, 1L, 1L, 8L, 9L, 0L, 1L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(7747, 0), (7748, 10927), (7749, 10928)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2382 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 10929) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [10927, 10928, 10929] |
| reports | [(10927, 7747), (10928, 7748), (10929, 7749)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 12, 1: 4, 2: 2, 3: 0, 4: 4, 5: 0, 6: 0, 7: 1, 8: 3, 9: 3, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 1L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeeeeeeefgggggggggggggg=gseeee@ggggg=g=gsee@gggshg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(10000, 11190), (12382, 13572), (13573, 14763), (14764, 14999)] |
| AllRedDataSegments | [(0, 1191), (2383, 3573), (4765, 5955), (5956, 7146), (7147, 8337), (9529, 9999), (3574, 4764), (8338, 9527), (9528, 9528), (1192, 2381), (2382, 2382)] |
| AmountGreenDataReceivedByApp | 3809 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 14999 |
| EORP | 9999 |
| GreenSegmentsReceived | [(10000, 11190), (12382, 13572), (13573, 14763), (14764, 14999)] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 7747 |
| InitialReportSerialNumber | 10927 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'GreenPartSegmentArrival', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(7747, 0), (7748, 10927), (7749, 10928)] |
| allReports | [(10927, 7747), (10928, 7748), (10928, 7748), (10928, 7748), (10929, 7749)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 4L, 4L, 4L, 7L, 2L, 8L, 9L, 0L, 0L, 1L, 8L, 8L, 8L, 9L, 0L, 1L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(7747, 0), (7748, 10927), (7749, 10928)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 2382 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 10929) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [10927, 10928, 10929] |
| reports | [(10927, 7747), (10928, 7748), (10929, 7749)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 2, 2: 1, 3: 0, 4: 3, 5: 0, 6: 0, 7: 1, 8: 5, 9: 3, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 1L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-27 14:36:58,892 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-27 14:37:06,315 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-10-27 14:37:14,562 - pyltprecv - INFO - GreenPartReception.indication (16:2) len(1191) EOB(0) md5(df2d55384ca1bcb5362bd5480cbb708d)

\*\*\* 2014-10-27 14:37:16,536 - pyltprecv - INFO - GreenPartReception.indication (16:2) len(1191) EOB(0) md5(98fe31cdecc45f0d822e632198c97702)

\*\*\* 2014-10-27 14:37:17,505 - pyltprecv - INFO - GreenPartReception.indication (16:2) len(1191) EOB(0) md5(8bac3f65a5ffa7d1c8b8a1052a9ceb14)

\*\*\* 2014-10-27 14:37:18,523 - pyltprecv - INFO - GreenPartReception.indication (16:2) len(236) EOB(1) md5(ba8c0f7de455764561622776b8883d9a)

\*\*\* 2014-10-27 14:37:18,524 - pyltprecv - INFO - Green Data Extents received for session (16, 2): (10000, 1191) (12382, 2618)

\*\*\* 2014-10-27 14:37:34,377 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 2) len(10000) EOB(0) md5(dc50add066871756c3f0260f0aa76cd2)

### Sender PCAP

14:37:05.782294 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:37:06.765502 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

14:37:07.748199 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

14:37:08.731197 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

14:37:09.714171 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

14:37:10.696897 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

14:37:11.679558 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7147, 1191)

14:37:12.662425 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8338, 1191)

14:37:13.645159 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9529, 471) CP(7747)

14:37:14.054497 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(10000, 1191)

14:37:15.037259 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(11191, 1191)

14:37:16.020064 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(12382, 1191)

14:37:17.002904 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(13573, 1191)

14:37:17.986239 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x7 Green data, EOB C(6) Data(14764, 236)

14:37:18.780093 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9529, 471) CP(7747)

14:37:19.803196 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7747, 10927) [0 , 10000]

14:37:19.805972 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(10927)

14:37:19.833300 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

14:37:20.816078 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

14:37:21.798949 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8338, 1190)

14:37:22.780946 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(9528, 1) CP(7748)

14:37:27.781735 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(9528, 1) CP(7748)

14:37:32.782783 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(9528, 1) CP(7748)

14:37:32.819067 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7748, 10928) [0 , 10000]

14:37:32.821326 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(10928)

14:37:32.848697 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1190)

14:37:33.830733 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(2382, 1) CP(7749)

14:37:34.891170 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7749, 10929) [0 , 10000]

14:37:34.892585 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(10929)

### Receiver PCAP

14:37:06.282351 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:37:08.248244 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

14:37:10.214251 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

14:37:11.196953 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

14:37:12.179608 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7147, 1191)

14:37:14.554558 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(10000, 1191)

14:37:16.520128 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(12382, 1191)

14:37:17.502973 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(13573, 1191)

14:37:18.486346 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x7 Green data, EOB C(6) Data(14764, 236)

14:37:19.280162 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9529, 471) CP(7747)

14:37:19.303118 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7747, 10927) [0 , 10000]

14:37:20.306025 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(10927)

14:37:21.316123 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

14:37:22.299006 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8338, 1190)

14:37:28.281804 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(9528, 1) CP(7748)

14:37:28.316051 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7748, 10928) [0 , 10000]

14:37:30.317920 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7748, 10928) [0 , 10000]

14:37:32.319032 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7748, 10928) [0 , 10000]

14:37:33.321396 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(10928)

14:37:33.348746 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1190)

14:37:34.330950 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(2382, 1) CP(7749)

14:37:34.391130 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(7749, 10929) [0 , 10000]

14:37:35.392637 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(10929)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 --rate 1000 --segmentSize 1200 \

-r file:redPart \

-g file:greenPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(10000, 11184), (11185, 12369), (12370, 13554), (13555, 14739), (14740, 14999)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 5925), (5926, 7110), (7111, 8295), (8296, 9480), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (1186, 2370), (5926, 7110), (7111, 8295), (7111, 8295)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 14999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 15417 |
| InitialReportSerialNumber | 3519 |
| MiscoloredData | False |
| ReceiveIndications | ['GreenPartSegmentArrival', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(15417, 0), (15417, 0), (15417, 0), (15417, 0), (15417, 0), (15417, 0), (15417, 0), (15418, 0), (15418, 0)] |
| allReports | [(3519, 15417), (3520, 15418)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 2L, 4L, 4L, 4L, 4L, 7L, 2L, 2L, 2L, 2L, 2L, 2L, 8L, 9L, 0L, 0L, 1L, 1L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(15417, 0), (15418, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 8295 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 3520) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [3519, 3520] |
| reports | [(3519, 15417), (3520, 15418)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 10, 1: 2, 2: 7, 3: 0, 4: 4, 5: 0, 6: 0, 7: 1, 8: 2, 9: 2, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 1L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 78 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414508167.227910 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414508167.228054 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414508167.229490 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414508182.527371 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414508198.338425 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(10000, 11184), (11185, 12369), (12370, 13554), (13555, 14739), (14740, 14999)] |
| AllRedDataSegments | [(0, 1185), (2371, 3555), (3556, 4740), (4741, 5925), (8296, 9480), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (1186, 2370), (5926, 7110), (7111, 8295)] |
| AmountGreenDataReceivedByApp | 5000 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 14999 |
| EORP | 9999 |
| GreenSegmentsReceived | [(10000, 11184), (11185, 12369), (12370, 13554), (13555, 14739), (14740, 14999)] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 15417 |
| InitialReportSerialNumber | 3519 |
| MiscoloredData | False |
| ReceiveIndications | ['GreenPartSegmentArrival', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(15417, 0), (15417, 0), (15417, 0), (15417, 0), (15417, 0), (15418, 0)] |
| allReports | [(3519, 15417), (3519, 15417), (3519, 15417), (3520, 15418)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 4L, 4L, 4L, 4L, 7L, 2L, 8L, 2L, 2L, 8L, 2L, 2L, 8L, 9L, 0L, 0L, 1L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(15417, 0), (15418, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 8295 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 3520) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [3519, 3520] |
| reports | [(3519, 15417), (3520, 15418)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 7, 1: 1, 2: 5, 3: 0, 4: 4, 5: 0, 6: 0, 7: 1, 8: 4, 9: 2, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 1L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 78 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sssssssssssgss+gss+gsssstgs

### Sender PCAP

10:56:08.437906 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

10:56:09.645400 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

10:56:10.853215 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

10:56:12.060448 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

10:56:13.268253 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4741, 1185)

10:56:14.475877 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5926, 1185)

10:56:15.682476 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7111, 1185)

10:56:16.889309 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8296, 1185)

10:56:17.393148 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:18.600987 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) x04 Green data, NOT EOB C(6) Data(10000, 1185)

10:56:19.809308 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) x04 Green data, NOT EOB C(6) Data(11185, 1185)

10:56:21.016586 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) x04 Green data, NOT EOB C(6) Data(12370, 1185)

10:56:22.224704 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) x04 Green data, NOT EOB C(6) Data(13555, 1185)

10:56:22.529577 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x7 Green data, EOB C(6) Data(14740, 260)

10:56:23.033075 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:25.036104 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:27.041656 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:29.044835 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:31.050542 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:33.054845 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:33.646737 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment RS(15417, 3519) [0 , 10000]

10:56:33.699510 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x9 Report-acknowledgment segment RA(3519)

10:56:33.708986 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

10:56:34.095684 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5926, 1185)

10:56:35.304290 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(7111, 1185) CP(15418)

10:56:37.307265 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(7111, 1185) CP(15418)

10:56:38.312328 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment RS(15418, 3520) [0 , 8296]

10:56:38.339338 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x9 Report-acknowledgment segment RA(3520)

### Receiver PCAP

10:56:08.937973 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

10:56:11.353317 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

10:56:12.560585 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

10:56:13.768330 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4741, 1185)

10:56:17.389367 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8296, 1185)

10:56:19.101033 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) x04 Green data, NOT EOB C(6) Data(10000, 1185)

10:56:20.309376 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) x04 Green data, NOT EOB C(6) Data(11185, 1185)

10:56:21.516648 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) x04 Green data, NOT EOB C(6) Data(12370, 1185)

10:56:22.724772 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) x04 Green data, NOT EOB C(6) Data(13555, 1185)

10:56:23.029638 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x7 Green data, EOB C(6) Data(14740, 260)

10:56:23.533140 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:23.536811 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment RS(15417, 3519) [0 , 10000]

10:56:25.536168 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:27.541716 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:28.145463 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment RS(15417, 3519) [0 , 10000]

10:56:29.544905 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:31.550601 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(9481, 519) CP(15417)

10:56:33.146666 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment RS(15417, 3519) [0 , 10000]

10:56:34.199564 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x9 Report-acknowledgment segment RA(3519)

10:56:34.209029 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

10:56:34.595746 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5926, 1185)

10:56:37.807333 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(7111, 1185) CP(15418)

10:56:37.812264 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment RS(15418, 3520) [0 , 8296]

10:56:38.839400 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x9 Report-acknowledgment segment RA(3520)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 1 , 2 , 4 , 7 , 8 , 9 | Good: Segment Types: 0 , 1 , 2 , 4 , 7 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 1 , 2 , 4 , 7 , 8 , 9 | Good: Segment Types: 0 , 1 , 2 , 4 , 7 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  | Ensure: Multiple Reports Sent. | Good: RSN RSN\_CSN [(10927, 7747), (10928, 7748), (10928, 7748), (10928, 7748), (10929, 7749)] | Good: RSN RSN\_CSN [(3519, 15417), (3519, 15417), (3519, 15417), (3520, 15418)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  | Ensure: Multiple Report acknowledgements sent | Good: Report Acknowledgement(s) sent: [10927, 10928, 10929] | Good: Report Acknowledgement(s) sent: [3519, 3520] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(7747, 0), (7747, 0), (7748, 10927), (7748, 10927), (7748, 10927), (7749, 10928)] | Good: Checkpoint transmitted: [(15417, 0), (15417, 0), (15417, 0), (15417, 0), (15417, 0), (15417, 0), (15417, 0), (15418, 0), (15418, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | Good: StartReportSegment Timer (Report(s) retransmitted) | Good: StartReportSegment Timer (Report(s) retransmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  | Ensure: Checkpoints retransmitted | Good: checkpoints retransmitted: [(7747, 2, 10927), (7748, 3, 10928), (7749, 1, 10929)] | Good: checkpoints retransmitted: [(15417, 7, 3519), (15418, 2, 3520)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | Good: Report retransmitted | Good: Report retransmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No cancel due to RLEXC | No cancel due to RLEXC |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  | Ensure: GreenPartReception.indication | Good: GreenPartReception.indication | Good: GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  | Ensure: SendReceptionReport | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  | Look for retransmitted data. | Good: Data retransmitted. | Good: Data retransmitted. |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report (with same RSN) after report-ack for that RSN. | Good: no report (with same RSN) after report-ack for that RSN. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (78) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (7747) | Good: InitialCheckpointSerialNumber OK (15417) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 7747 | Good: InitialCheckpointSerialNumber: 15417 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Corrupt

Send a malformed segment (e.g. bad version #) in the middle of a red data transmission.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 10000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

During an LTP transfer session with only red data, send a malformed segment then complete the session.

## Test-Specific Success Criteria

No LTP segment is generated by the receiver in response to the malformed segment (malformed segment is silently discarded); data transfer succeeds.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 -f corrupt

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146), (7147, 8337), (8338, 9528), (9529, 9999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2930 |
| InitialReportSerialNumber | 13548 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2930, 0)] |
| allReports | [(13548, 2930)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(2930, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 13548) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [13548] |
| reports | [(13548, 2930)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeefgggggggggshg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146), (7147, 8337), (8338, 9528), (9529, 9999)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2930 |
| InitialReportSerialNumber | 13548 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2930, 0)] |
| allReports | [(13548, 2930)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(2930, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [('Bad Version:', 1L)] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 13548) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [13548] |
| reports | [(13548, 2930)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 9, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 9L] |
| segmentVersions | [0L, 1L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-11-03 14:10:25,306 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-11-03 14:10:32,785 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-11-03 14:10:40,641 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(10000) EOB(1) md5(dc50add066871756c3f0260f0aa76cd2)

### Sender PCAP

14:10:32.244676 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:10:33.227116 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

14:10:34.210072 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

14:10:35.192718 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

14:10:36.175570 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

14:10:37.158298 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

14:10:38.141057 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7147, 1191)

14:10:39.123966 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8338, 1191)

14:10:40.107294 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(2930)

14:10:41.150615 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(2930, 13548) [0 , 10000]

14:10:41.152210 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(13548)

### Receiver PCAP

14:10:32.744733 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:10:33.534969 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv1L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:10:33.727206 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

14:10:34.710144 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

14:10:35.692780 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

14:10:36.675621 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

14:10:37.658340 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

14:10:38.641109 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7147, 1191)

14:10:39.624030 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8338, 1191)

14:10:40.607364 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(2930)

14:10:40.650571 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(2930, 13548) [0 , 10000]

14:10:41.652251 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(13548)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 -f corrupt

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 5925), (5926, 7110), (7111, 8295), (8296, 9480), (9481, 9999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4586 |
| InitialReportSerialNumber | 560 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4586, 0)] |
| allReports | [(560, 4586)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(4586, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 560) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [560] |
| reports | [(560, 4586)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 37 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414089649.486197 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 37)

\*\*\* 1414089649.486458 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414089649.487975 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414089659.663391 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 37)

\*\*\* 1414089660.713992 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 37)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 5925), (5926, 7110), (7111, 8295), (8296, 9480), (9481, 9999)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4586 |
| InitialReportSerialNumber | 560 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4586, 0)] |
| allReports | [(560, 4586)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(4586, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [('Bad Version:', 1L)] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 560) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [560] |
| reports | [(560, 4586)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 9, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 9L] |
| segmentVersions | [0L, 1L] |
| senderCancelled | False |
| sessionNumber | 37 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

Running ionadmin with cmdFileName: node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

logSize: 0

logKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

ssssssssstgs

### Sender PCAP

14:40:50.696379 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

14:40:51.909328 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

14:40:53.116672 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

14:40:54.324146 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

14:40:55.531354 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4741, 1185)

14:40:56.738559 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5926, 1185)

14:40:57.946013 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7111, 1185)

14:40:59.154659 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8296, 1185)

14:40:59.660557 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4586)

14:41:00.667617 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x8 Report segment RS(4586, 560) [0 , 10000]

14:41:00.709028 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x9 Report-acknowledgment segment RA(560)

### Receiver PCAP

14:40:51.196443 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

14:40:51.985835 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv1L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

14:40:52.409396 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

14:40:53.616745 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

14:40:54.824209 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

14:40:56.031433 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4741, 1185)

14:40:57.238619 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5926, 1185)

14:40:58.446075 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7111, 1185)

14:40:59.654727 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8296, 1185)

14:41:00.160614 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4586)

14:41:00.167552 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x8 Report segment RS(4586, 560) [0 , 10000]

14:41:01.209095 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 37) 0x9 Report-acknowledgment segment RA(560)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 3 , 8 , 9 | Good: Segment Types: 0 , 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 3 , 8 , 9 | Good: Segment Types: 0 , 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(13548, 2930)] | Good: RSN RSN\_CSN [(560, 4586)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [13548] | Good: Report Acknowledgement(s) sent: [560] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  | Check for segment with LTP version #1 and ensure that it is ignored. | Good: verified by hand. | Good: verified by hand. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(2930, 0)] | Good: Checkpoint transmitted: [(4586, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No checkpoints transmitted | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (37) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (2930) | Good: InitialCheckpointSerialNumber OK (4586) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 2930 | Good: InitialCheckpointSerialNumber: 4586 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Unreach1

Sender transmitting red data to an unreachable destination client service ID but receiver has queue to sender.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 5000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Attempt to open an LTP connection to an LTP client service that is not extant at the receiver.

## Test-Specific Success Criteria

Receiver responds with an an LTP cancel segment with reason code: UNREACH to the sender. Sender closes the connection on receipt of the UNREACH segment.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 3 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 14L, 15L, 14L] |
| cancelReasonCodes | [1, 1] |
| checkpoints | [] |
| clientServiceID | 3 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 2382 |
| lastCancelOrAck | 14 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 2, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 2, 15: 1} |
| segmentTypes | [0L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeefggs]gs

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [0L] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 14L, 0L, 14L, 15L] |
| cancelReasonCodes | [1, 1] |
| checkpoints | [] |
| clientServiceID | 3 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 2382 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 2, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 2, 15: 1} |
| segmentTypes | [0L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-28 11:20:39,529 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

### Sender PCAP

11:20:46.470284 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(0, 1192)

11:20:47.453161 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(1192, 1191)

11:20:47.504878 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(UNREACH)

11:20:48.436038 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xF Cancel-acknowledgment segment to block receiver

11:20:48.472584 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(UNREACH)

### Receiver PCAP

11:20:46.970381 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(0, 1192)

11:20:47.004812 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(UNREACH)

11:20:47.953218 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(1192, 1191)

11:20:47.972517 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(UNREACH)

11:20:48.936115 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xF Cancel-acknowledgment segment to block receiver

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

--destClientID 17 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 14L, 15L] |
| cancelReasonCodes | [1] |
| checkpoints | [] |
| clientServiceID | 17 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 1185 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 45 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414090134.617323 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 45)

\*\*\* 1414090134.617463 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414090134.618419 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414090136.883224 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 45) reason(1)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 14L, 15L] |
| cancelReasonCodes | [1] |
| checkpoints | [] |
| clientServiceID | 17 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 1185 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 1, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 45 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

Running ionadmin with cmdFileName: node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

logSize: 0

logKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sgs

### Sender PCAP

14:48:55.828080 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 45) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(17) Data(0, 1186)

14:48:56.831100 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 45) 0xE Cancel segment from block receiver Reason(UNREACH)

14:48:56.889879 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 45) 0xF Cancel-acknowledgment segment to block receiver

### Receiver PCAP

14:48:56.328133 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 45) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(17) Data(0, 1186)

14:48:56.331037 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 45) 0xE Cancel segment from block receiver Reason(UNREACH)

14:48:57.389946 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 45) 0xF Cancel-acknowledgment segment to block receiver

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 14 , 15 | Good: Segment Types: 0 , 14 , 15 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 14 , 15 | Good: Segment Types: 0 , 14 , 15 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length Data | Good: CSID Offset Length Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No Reports [] | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x01 | Good: CancelReasonCodes: 0x01 | Good: CancelReasonCodes: 0x01 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- Receiver has queue to sender. | No check -- Receiver has queue to sender. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  | Ensure: UNREACH sent | Good: UNREACH sent | Good: UNREACH sent |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | No checkpoints transmitted | No checkpoints transmitted |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No checkpoints transmitted | No checkpoints transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted . | No checkpoints retransmitted . |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | NO InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report sent. | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: transmission queue to receiver present | No check: transmission queue to receiver present |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  | Ensure: CancelACK to receiver | Good: CancelACK to receiver | Good: CancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  | Ensure: no cancel sent after cancel ack received. | Good: no cancel sent by receiver after cancel ack received. | Good: no cancel sent after cancel ack received. |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  | Ensure: not all 5k of red data are transmitted. | Good: Verified by inspection of pcap file. | Good: Verified by inspection of pcap file. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: No RedPartRectpion.indication | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: No TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | NO InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | NO ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (45) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | No checkpoints | No checkpoints |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | No checkpoints | No checkpoints |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Unreach2

Unreachable client service ID and receiver has no queue to sender.

## Settings

### ION to Python

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 5000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

### Python to ION

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 5000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

LTP sender sends to a receiver / client pair where there is not listening client and the receiver does not know how to transmit to the sender.

## Test-Specific Success Criteria

Sender eventually gives up and cancels the session. Receiver never transmits anything.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 3 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{22: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, }}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 4999 |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2608 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [2608] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(2608, 0)] |
| clientServiceID | 3 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 4999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 10, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeefggggg=g=g=g=g=g=g=g=g=g{gggggggggg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999), (4765, 4999)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 4999 |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2608 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [2608] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(2608, 0)] |
| clientServiceID | 3 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 4999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 10, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-24 07:59:50,992 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

### Sender PCAP

07:59:58.934436 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(0, 1192)

07:59:59.917251 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(1192, 1191)

08:00:00.900213 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(2383, 1191)

08:00:01.882597 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(3574, 1191)

08:00:02.865984 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:07.930376 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:12.931502 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:17.932590 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:22.934000 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:27.935301 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:32.936355 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:37.937617 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:42.938806 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:47.939510 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:52.942456 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:00:57.944359 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:02.945878 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:07.947309 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:12.948625 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:17.950191 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:22.951168 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:27.952506 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:32.953378 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:37.954734 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

### Receiver PCAP

07:59:59.434494 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(0, 1192)

08:00:00.417331 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(1192, 1191)

08:00:01.400252 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(2383, 1191)

08:00:02.382633 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(3) Data(3574, 1191)

08:00:03.366050 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:08.430424 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:13.431587 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:18.432666 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:23.434075 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:28.435378 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:33.436494 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:38.437675 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:43.438853 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:48.439590 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(3) Data(4765, 235) CP(2608)

08:00:53.442513 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:00:58.444412 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:03.446000 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:08.447401 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:13.448705 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:18.450273 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:23.451216 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:28.452581 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:33.453447 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

08:01:38.454829 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 22 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

--destClientID 17 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 8719 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [8719] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L] |
| cancelReasonCodes | [2] |
| checkpoints | [(8719, 0)] |
| clientServiceID | 17 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 67 |
| sessionOriginator | 22 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414152469.016252 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (22, 67)

\*\*\* 1414152469.016408 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414152469.018000 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414152469.226103 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (22, 67)

\*\*\* 1414152511.335978 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (22, 67) reason(2)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 8719 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [8719] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L] |
| cancelReasonCodes | [2] |
| checkpoints | [(8719, 0)] |
| clientServiceID | 17 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 67 |
| sessionOriginator | 22 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

ssssssssssssssssssssss

### Sender PCAP

08:07:49.223212 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:51.229567 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:53.233941 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:55.237339 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:57.243992 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:59.249510 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:01.253283 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:03.258599 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:05.264307 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:07.268910 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:09.274543 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:11.281290 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:13.289561 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:15.292583 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:17.298929 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:19.307672 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:21.309088 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:23.313941 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:25.322026 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:27.328737 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:29.337466 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:31.339607 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0xC Cancel segment from block sender Reason(RLEXC)

### Receiver PCAP

08:07:49.723268 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:51.729641 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:53.733999 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:55.737375 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:57.744111 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:07:59.749562 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:01.753351 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:03.758655 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:05.764372 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:07.768992 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:09.774605 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:11.781351 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:13.789674 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:15.792639 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:17.799010 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:19.807730 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:21.809161 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:23.814023 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:25.822078 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:27.828793 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:29.837538 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0x3 Red data, Checkpoint, EORP, EOB C(17) Data(0, 100) CP(8719)

08:08:31.839672 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(22, 67) 0xC Cancel segment from block sender Reason(RLEXC)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 3 , 12 | Good: Segment Types: 3 , 12 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 3 , 12 | Good: Segment Types: 3 , 12 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN Data | Good: CSID Offset Length CSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No Reports [] | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Verify that cancel reason code 0x01 is NOT present (reason code 0x02 should be) | Good: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  | Verify that no segments are sent by the receiver (no segment types 8, 13, 14) | Good: verified by hand. | Good: verified by hand. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No queue to sender. | No queue to sender. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0), (2608, 0)] | Good: Checkpoint transmitted: [(8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0), (8719, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(2608, 10, False)] | Good: checkpoints retransmitted: [(8719, 21, False)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report sent. | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  | Ensure: Cancel segment retransmission (segment type 12) | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  | Ensure that the session is cancelled with RLEXC | Good: Session cancelled with RLEXC | Good: Session cancelled with RLEXC |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  | Ensure: Cancel segment retransmission by sender (segment type 12). | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  | Ensure that this is so; receiver should not transmit cancel ACKs. | Good: no segment type 15 present. | Good: no segment type 15 present. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  | Ensure: no cancelACK to sender | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel ack received (was looking for 13 in response to 12 in [0L, 0L, 0L, 0L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L]) | No cancel ack received (was looking for 13 in response to 12 in [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L]) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting after it sends cancel(RLEXC) | Good: Sender stops transmitting after it sends cancel(RLEXC) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | NO ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (67) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (2608) | Good: InitialCheckpointSerialNumber OK (8719) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 2608 | Good: InitialCheckpointSerialNumber: 8719 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Miscolored1

Red segment in Green Space

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 5000 | (Bytes) |
| GREEN\_SIZE | 5000 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

After establishing a connection and transmitting the Red Part data (including the EORP segment), and beginning to send the Green Part, send a Red segment with an offset in the Green sequence space.

## Test-Specific Success Criteria

Receiver responds with an LTP cancel segment with reason code: MISCOLORED (and closes the connection). Sender cancels the session on receipt of the cancellation segment and sends a cancel ACK.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -r 5000 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 -T 4 -f miscolored

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6190), (6191, 7381)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 7780 |
| InitialReportSerialNumber | 10609 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(7780, 0)] |
| allReports | [(10609, 7780)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 4L, 8L, 9L, 4L, 14L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(7780, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | ('Ack', 10609) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [10609] |
| reports | [(10609, 7780)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 2, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 8L, 9L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeeefggggggshggs]g

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6190), (6191, 7381)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 4999), (5000, 6190)] |
| AmountGreenDataReceivedByApp | 1191 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | 4999 |
| GreenSegmentsReceived | [(5000, 6190)] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [9L, 4L] |
| InitialCheckpointSerialNumber | 7780 |
| InitialReportSerialNumber | 10609 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(7780, 0)] |
| allReports | [(10609, 7780)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 8L, 4L, 0L, 14L, 9L, 4L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(7780, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 6190 |
| lastCancelOrAck | 15 |
| lastReportOrAck | ('Ack', 10609) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [10609] |
| reports | [(10609, 7780)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 5, 1: 0, 2: 1, 3: 0, 4: 2, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 8L, 9L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-24 08:27:08,316 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-24 08:27:15,738 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-24 08:27:19,711 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(5000) EOB(0) md5(d00321d3eb2086742076e4386efbefb4)

\*\*\* 2014-10-24 08:27:19,931 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1191) EOB(0) md5(df2d55384ca1bcb5362bd5480cbb708d)

\*\*\* 2014-10-24 08:27:20,649 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 1) reason MISCOLORED

\*\*\* 2014-10-24 08:27:20,649 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 08:27:20,649 - pyltprecv - INFO - (0, 5000)

### Sender PCAP

08:27:15.231281 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

08:27:16.213717 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

08:27:17.196547 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

08:27:18.179366 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

08:27:19.162220 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(7780)

08:27:19.382639 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(5000, 1191)

08:27:20.218116 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(7780, 10609) [0 , 5000]

08:27:20.365361 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(10609)

08:27:20.392844 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(6191, 1191)

08:27:21.152615 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(MISCOLORED)

08:27:21.375601 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xF Cancel-acknowledgment segment to block receiver

### Receiver PCAP

08:27:15.731329 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

08:27:16.713771 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

08:27:17.696598 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

08:27:18.679409 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

08:27:19.662269 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(7780)

08:27:19.718072 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(7780, 10609) [0 , 5000]

08:27:19.882688 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(5000, 1191)

08:27:20.610819 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5000, 1191)

08:27:20.652575 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(MISCOLORED)

08:27:20.865400 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(10609)

08:27:20.892904 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(6191, 1191)

08:27:21.875814 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xF Cancel-acknowledgment segment to block receiver

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-g file:greenPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 -T 4 -f miscolored

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6184), (6185, 7369), (7370, 8554), (8555, 9739), (9740, 9999)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1757 |
| InitialReportSerialNumber | 5947 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1757, 0)] |
| allReports | [(5947, 1757)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 8L, 4L, 9L, 4L, 4L, 4L, 7L] |
| cancelReasonCodes | [] |
| checkpoints | [(1757, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 5947) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [5947] |
| reports | [(5947, 1757)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 4, 5: 0, 6: 0, 7: 1, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 62 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414153771.930870 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 62)

\*\*\* 1414153771.931008 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414153771.932040 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414153778.100491 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 62)

\*\*\* 1414153782.087902 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 62)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6184), (6185, 7369), (7370, 8554), (8555, 9739), (9740, 9999)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 4999), (5000, 6184)] |
| AmountGreenDataReceivedByApp | 5000 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 4999 |
| GreenSegmentsReceived | [(5000, 6184), (6185, 7369), (7370, 8554), (8555, 9739), (9740, 9999)] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1757 |
| InitialReportSerialNumber | 5947 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1757, 0)] |
| allReports | [(5947, 1757)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 8L, 4L, 9L, 0L, 4L, 4L, 4L, 7L] |
| cancelReasonCodes | [] |
| checkpoints | [(1757, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 6184 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 5947) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [5947] |
| reports | [(5947, 1757)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 5, 1: 0, 2: 1, 3: 0, 4: 4, 5: 0, 6: 0, 7: 1, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 62 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

ssssstgsssssss

### Sender PCAP

08:29:33.139648 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

08:29:34.345854 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

08:29:35.551402 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

08:29:36.757323 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

08:29:37.060043 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(1757)

08:29:38.062824 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x8 Report segment RS(1757, 5947) [0 , 5000]

08:29:38.269176 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) x04 Green data, NOT EOB C(6) Data(5000, 1185)

08:29:38.270611 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x9 Report-acknowledgment segment RA(5947)

08:29:39.475045 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) x04 Green data, NOT EOB C(6) Data(6185, 1185)

08:29:40.681617 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) x04 Green data, NOT EOB C(6) Data(7370, 1185)

08:29:41.886591 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) x04 Green data, NOT EOB C(6) Data(8555, 1185)

08:29:42.087668 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x7 Green data, EOB C(6) Data(9740, 260)

### Receiver PCAP

08:29:33.639713 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

08:29:34.845932 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

08:29:36.051464 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

08:29:37.257401 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

08:29:37.560089 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(1757)

08:29:37.562747 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x8 Report segment RS(1757, 5947) [0 , 5000]

08:29:38.769249 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) x04 Green data, NOT EOB C(6) Data(5000, 1185)

08:29:38.770642 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x9 Report-acknowledgment segment RA(5947)

08:29:39.499969 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5000, 1185)

08:29:39.975094 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) x04 Green data, NOT EOB C(6) Data(6185, 1185)

08:29:41.181660 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) x04 Green data, NOT EOB C(6) Data(7370, 1185)

08:29:42.386652 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) x04 Green data, NOT EOB C(6) Data(8555, 1185)

08:29:42.587734 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 62) 0x7 Green data, EOB C(6) Data(9740, 260)

## Comments:

### Python to ION Comments

ION LTP disregards the rogue Red segment since it is received after the report-ack for the legitimate red-part of the block is processed. Scott argues that this is within the letter of the spec (6.21 says that the receiver SHOULD [not MUST] enqueue a CR segment with reason code MISCOLORED).  
  
Python LTP receiver issues a CANCEL segment with reason code MISCOLORED because the red segment is received before the EOB.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 2 , 4 , 8 , 9 , 14 , 15 | Good: Segment Types: 0 , 2 , 4 , 7 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 2 , 4 , 8 , 9 , 14 , 15 | Good: Segment Types: 0 , 2 , 4 , 7 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(10609, 7780)] | Good: RSN RSN\_CSN [(5947, 1757)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [10609] | Good: Report Acknowledgement(s) sent: [5947] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x03 | Good: CancelReasonCodes: 0x03 | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(7780, 0)] | Good: Checkpoint transmitted: [(1757, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | Good: GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segment retransmission | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segment retransmission | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  | Ensure: CancelACK to receiver | Good: CancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | Good: no cancel sent after cancel ack received. | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting data after cancel. | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (62) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (7780) | Good: InitialCheckpointSerialNumber OK (1757) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 7780 | Good: InitialCheckpointSerialNumber: 1757 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Miscolored2

Green segment in Red space

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 5000 | (Bytes) |
| GREEN\_SIZE | 10000 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

After establishing a connection and transmitting the Red Part data (including the EORP segment), send a Green segment with an offset in the Red sequence space.

## Test-Specific Success Criteria

Receiver responds with an LTP cancel segment with reason code: MISCOLORED (and closes the connection). Sender cancels the session on receipt of the cancellation segment.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -r 5000 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 -S 2 --delay 1.5 -T 0 -f miscolored

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6190), (6191, 7381)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2820 |
| InitialReportSerialNumber | 12559 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2820, 0)] |
| allReports | [(12559, 2820)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 4L, 8L, 9L, 4L, 14L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(2820, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | ('Ack', 12559) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [12559] |
| reports | [(12559, 2820)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 2, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 8L, 9L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeeeeeeefggggggshggs]g

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6190), (2383, 3573), (6191, 7381)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 4999)] |
| AmountGreenDataReceivedByApp | 1191 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | 4999 |
| GreenSegmentsReceived | [(5000, 6190)] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [9L, 4L] |
| InitialCheckpointSerialNumber | 2820 |
| InitialReportSerialNumber | 12559 |
| MiscoloredData | True |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2820, 0)] |
| allReports | [(12559, 2820)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 8L, 4L, 4L, 14L, 9L, 4L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(2820, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | ('Ack', 12559) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [12559] |
| reports | [(12559, 2820)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 3, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 8L, 9L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-24 10:04:26,963 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-24 10:04:34,440 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-10-24 10:04:38,378 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 2) len(5000) EOB(0) md5(d00321d3eb2086742076e4386efbefb4)

\*\*\* 2014-10-24 10:04:38,600 - pyltprecv - INFO - GreenPartReception.indication (16:2) len(1191) EOB(0) md5(df2d55384ca1bcb5362bd5480cbb708d)

\*\*\* 2014-10-24 10:04:38,625 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason MISCOLORED

\*\*\* 2014-10-24 10:04:38,625 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 10:04:38,625 - pyltprecv - INFO - (0, 5000)

### Sender PCAP

10:04:33.917366 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

10:04:34.900295 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

10:04:35.887423 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

10:04:36.870520 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

10:04:37.853578 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(2820)

10:04:38.074196 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(5000, 1191)

10:04:38.886059 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2820, 12559) [0 , 5000]

10:04:39.056900 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(12559)

10:04:39.084461 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(6191, 1191)

10:04:39.130687 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(MISCOLORED)

10:04:40.067522 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xF Cancel-acknowledgment segment to block receiver

### Receiver PCAP

10:04:34.417433 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

10:04:35.400358 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

10:04:36.387500 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

10:04:37.370592 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

10:04:38.353679 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(2820)

10:04:38.385993 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2820, 12559) [0 , 5000]

10:04:38.574273 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(5000, 1191)

10:04:38.614742 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(2383, 1191)

10:04:38.630610 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(MISCOLORED)

10:04:39.556972 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(12559)

10:04:39.584546 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(6191, 1191)

10:04:40.567600 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xF Cancel-acknowledgment segment to block receiver

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-g file:greenPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 -S 2 --delay 1.5 -T 0 -f miscolored

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6184), (6185, 7369), (7370, 8554), (8555, 9739), (9740, 10924), (10925, 12109), (12110, 13294), (13295, 14479), (14480, 14999)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 14999 |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 6942 |
| InitialReportSerialNumber | 4562 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(6942, 0)] |
| allReports | [(4562, 6942)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 8L, 4L, 9L, 4L, 4L, 4L, 4L, 4L, 4L, 4L, 7L] |
| cancelReasonCodes | [] |
| checkpoints | [(6942, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 14999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 4562) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [4562] |
| reports | [(4562, 6942)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 8, 5: 0, 6: 0, 7: 1, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 39 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414159724.236887 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 39)

\*\*\* 1414159724.237101 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414159724.238132 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414159730.407117 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 39)

\*\*\* 1414159739.559727 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 39)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(2371, 3555), (5000, 6184), (6185, 7369), (7370, 8554), (8555, 9739), (9740, 10924), (10925, 12109), (12110, 13294), (13295, 14479), (14480, 14999)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 4999)] |
| AmountGreenDataReceivedByApp | 11185 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 14999 |
| EORP | 4999 |
| GreenSegmentsReceived | [(2371, 3555), (5000, 6184), (6185, 7369), (7370, 8554), (8555, 9739), (9740, 10924), (10925, 12109), (12110, 13294), (13295, 14479), (14480, 14999)] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 6942 |
| InitialReportSerialNumber | 4562 |
| MiscoloredData | True |
| ReceiveIndications | ['RedPartReception', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival', 'GreenPartSegmentArrival'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(6942, 0)] |
| allReports | [(4562, 6942)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 8L, 4L, 4L, 9L, 4L, 4L, 4L, 4L, 4L, 4L, 4L, 7L] |
| cancelReasonCodes | [] |
| checkpoints | [(6942, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 14999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 4562) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [4562] |
| reports | [(4562, 6942)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 9, 5: 0, 6: 0, 7: 1, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 2L, 4L, 7L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 39 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

ssssstgsssssssssss

### Sender PCAP

10:08:45.449028 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

10:08:46.655739 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

10:08:47.866213 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

10:08:49.075326 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

10:08:49.389059 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(6942)

10:08:50.393551 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x8 Report segment RS(6942, 4562) [0 , 5000]

10:08:50.593729 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(5000, 1185)

10:08:50.595403 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x9 Report-acknowledgment segment RA(4562)

10:08:51.803313 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(6185, 1185)

10:08:53.012670 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(7370, 1185)

10:08:54.220985 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(8555, 1185)

10:08:55.429107 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(9740, 1185)

10:08:56.637583 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(10925, 1185)

10:08:57.844974 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(12110, 1185)

10:08:59.052908 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(13295, 1185)

10:08:59.557100 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x7 Green data, EOB C(6) Data(14480, 520)

### Receiver PCAP

10:08:45.949089 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

10:08:47.155822 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

10:08:48.366278 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

10:08:49.575378 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

10:08:49.889127 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(6942)

10:08:49.893496 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x8 Report segment RS(6942, 4562) [0 , 5000]

10:08:50.607583 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(2371, 1185)

10:08:51.093872 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(5000, 1185)

10:08:51.095443 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x9 Report-acknowledgment segment RA(4562)

10:08:52.303368 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(6185, 1185)

10:08:53.512788 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(7370, 1185)

10:08:54.721072 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(8555, 1185)

10:08:55.929167 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(9740, 1185)

10:08:57.137650 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(10925, 1185)

10:08:58.345061 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(12110, 1185)

10:08:59.552977 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) x04 Green data, NOT EOB C(6) Data(13295, 1185)

10:09:00.057180 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 39) 0x7 Green data, EOB C(6) Data(14480, 520)

## Comments:

### Python to ION Comments

ION LTPdisregards the rogue Green segment since it is received after the report-ack for the legitimate red-part of the block is processed. Scott argues that this is within the letter of the spec (6.21 says that the receiver SHOULD [not MUST] enqueue a CR segment with reason code MISCOLORED).  
  
Python LTP receiver issues a CANCEL segment with reason code MISCOLORED because the green segment is in the red space.  
  
ION delivered the correct red-part, then delivered the miscolored Green data to the application with the offset in the red space of the block.  
  
Later versions of ION correct this problem. It was not tested here because the ION to Python test is enough to cover the interoperability requirement.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 2 , 4 , 8 , 9 , 14 , 15 | Good: Segment Types: 0 , 2 , 4 , 7 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 2 , 4 , 8 , 9 , 14 , 15 | Good: Segment Types: 0 , 2 , 4 , 7 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(12559, 2820)] | Good: RSN RSN\_CSN [(4562, 6942)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [12559] | Good: Report Acknowledgement(s) sent: [4562] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x03 | Good: CancelReasonCodes: 0x03 | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(2820, 0)] | Good: Checkpoint transmitted: [(6942, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | Good: GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segment retransmission | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segment retransmission | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  | Ensure: CancelACK to receiver | Good: CancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | Good: no cancel sent after cancel ack received. | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting data after cancel. | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  | Ensure: miscolored data present | Good: miscolored data present | Good: miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (39) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (2820) | Good: InitialCheckpointSerialNumber OK (6942) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 2820 | Good: InitialCheckpointSerialNumber: 6942 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Miscolored3

Red segment in Green Space after blocking initial EORP.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 5000 | (Bytes) |
| GREEN\_SIZE | 5000 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Block the initial EORP, start sending Green segments, then put a red segment in the green space.

## Test-Specific Success Criteria

Receiver responds with an LTP cancel segment with reason code: MISCOLORED (and closes the connection). Sender cancels the session on receipt of the cancellation segment.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -r 5000 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 -T 4 --skip 1 --mungeOffset -1000 -f miscolored

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6190), (6191, 7381), (7382, 8572)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 785 |
| InitialReportSerialNumber | 7331 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(785, 0)] |
| allReports | [(7331, 785)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 4L, 8L, 9L, 4L, 4L, 14L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(785, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | ('Ack', 7331) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [7331] |
| reports | [(7331, 785)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 3, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 8L, 9L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeeefggggggsee@gggs]g

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6190), (6191, 7381), (7382, 8572)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (4765, 4999), (5191, 6381)] |
| AmountGreenDataReceivedByApp | 2382 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | 4999 |
| GreenSegmentsReceived | [(5000, 6190), (6191, 7381)] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [4L] |
| InitialCheckpointSerialNumber | 785 |
| InitialReportSerialNumber | 7331 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(785, 0)] |
| allReports | [(7331, 785)] |
| allSegmentsInOrder | [0L, 0L, 0L, 2L, 8L, 4L, 9L, 4L, 0L, 14L, 4L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(785, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 6381 |
| lastCancelOrAck | 15 |
| lastReportOrAck | ('Ack', 7331) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [7331] |
| reports | [(7331, 785)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 3, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 8L, 9L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-24 10:12:26,662 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-24 10:12:34,136 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-10-24 10:12:38,319 - pyltprecv - INFO - GreenPartReception.indication (16:2) len(1191) EOB(0) md5(df2d55384ca1bcb5362bd5480cbb708d)

\*\*\* 2014-10-24 10:12:39,291 - pyltprecv - INFO - GreenPartReception.indication (16:2) len(1191) EOB(0) md5(6b98158cad4c21e5af65e568b0fd432e)

\*\*\* 2014-10-24 10:12:40,060 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason MISCOLORED

\*\*\* 2014-10-24 10:12:40,060 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 10:12:40,060 - pyltprecv - INFO - (0, 3574) (4765, 235)

### Sender PCAP

10:12:33.617176 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

10:12:34.600087 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

10:12:35.583061 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

10:12:36.565881 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

10:12:37.548888 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(785)

10:12:37.769442 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(5000, 1191)

10:12:38.605087 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(785, 7331) [0 , 5000]

10:12:38.752089 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(7331)

10:12:38.779541 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(6191, 1191)

10:12:39.762607 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(7382, 1191)

10:12:40.565495 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(MISCOLORED)

10:12:40.745656 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xF Cancel-acknowledgment segment to block receiver

### Receiver PCAP

10:12:34.117255 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

10:12:35.100169 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

10:12:36.083132 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

10:12:38.048959 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(785)

10:12:38.105014 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(785, 7331) [0 , 5000]

10:12:38.269512 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(5000, 1191)

10:12:39.252192 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(7331)

10:12:39.279613 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(6191, 1191)

10:12:40.020174 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5191, 1191)

10:12:40.065440 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(MISCOLORED)

10:12:40.262681 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) x04 Green data, NOT EOB C(6) Data(7382, 1191)

10:12:41.245732 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xF Cancel-acknowledgment segment to block receiver

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-g file:greenPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 -T 4 -f miscolored

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6184), (6185, 7369)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3132 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['GreenPartSegmentArrival', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [3132] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(3132, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 4L, 4L, 14L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(3132, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 1, 3: 0, 4: 2, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 77 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414160182.763499 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 77)

\*\*\* 1414160182.763653 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414160182.765223 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414160190.879271 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 77) reason(3)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6184), (6185, 7369)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (5000, 6184)] |
| AmountGreenDataReceivedByApp | 2370 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [(5000, 6184), (6185, 7369)] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [4L] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['GreenPartSegmentArrival', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 4L, 0L, 14L, 4L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 6184 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 0, 3: 0, 4: 2, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 4L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 77 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sssss[gss

### Sender PCAP

10:16:23.974853 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

10:16:25.182513 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

10:16:26.389703 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

10:16:27.603148 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

10:16:27.906263 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(3132)

10:16:29.112425 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) x04 Green data, NOT EOB C(6) Data(5000, 1185)

10:16:30.320621 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) x04 Green data, NOT EOB C(6) Data(6185, 1185)

10:16:30.853944 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0xE Cancel segment from block receiver Reason(MISCOLORED)

10:16:30.885519 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0xF Cancel-acknowledgment segment to block receiver

### Receiver PCAP

10:16:24.474939 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

10:16:25.682598 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

10:16:26.889856 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

10:16:29.612488 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) x04 Green data, NOT EOB C(6) Data(5000, 1185)

10:16:30.350676 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5000, 1185)

10:16:30.353878 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0xE Cancel segment from block receiver Reason(MISCOLORED)

10:16:30.820713 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) x04 Green data, NOT EOB C(6) Data(6185, 1185)

10:16:31.385591 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 77) 0xF Cancel-acknowledgment segment to block receiver

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 2 , 4 , 8 , 9 , 14 , 15 | Good: Segment Types: 0 , 4 , 14 , 15 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 2 , 4 , 8 , 9 , 14 , 15 | Good: Segment Types: 0 , 4 , 14 , 15 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(7331, 785)] | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [7331] | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x03 | Good: CancelReasonCodes: 0x03 | Good: CancelReasonCodes: 0x03 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(785, 0)] | Good: Checkpoint transmitted: [(3132, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | Good: GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | NO InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. | No check: both sides have transmission queues. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  | Ensure: CancelACK to receiver | Good: CancelACK to receiver | Good: CancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | Good: no cancel sent after cancel ack received. | Good: no cancel sent after cancel ack received. |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting data after cancel. | Good: Sender stops transmitting data after cancel. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | NO InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (77) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (785) | Good: InitialCheckpointSerialNumber OK (3132) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 785 | Good: InitialCheckpointSerialNumber: 3132 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Miscolored4

Green segment in known Red space.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 5000 | (Bytes) |
| GREEN\_SIZE | 5000 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Get a green segment in the KNOWN RED space and before the EORP segment. Use ebtables to block the receipt of the EORP so that the green segment shows up in the known red space before the Red part of the block is completely received.

## Test-Specific Success Criteria

Cancel with reason code: MISCOLORD.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -r 5000 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

tcpdump -i lo -w python\_UDP.pcap &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 --skip 3 -T 0 --delay 2 -f miscolored

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6190), (6191, 7381), (7382, 8572), (8573, 9763), (9764, 9999)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 4999), (4765, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 295 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'GreenPartSegmentArrival', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [295] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(295, 0), (295, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 4L, 4L, 4L, 4L, 7L, 2L, 14L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(295, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 4999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 2, 3: 0, 4: 4, 5: 0, 6: 0, 7: 1, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 7L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeeefgggggggggg=gs]g

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(6191, 7381), (3574, 4764), (7382, 8572), (8573, 9763), (9764, 9999)] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (4765, 4999)] |
| AmountGreenDataReceivedByApp | 5000 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 4999 |
| GreenSegmentsReceived | [(6191, 7381), (3574, 4764), (7382, 8572), (8573, 9763), (9764, 9999)] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 295 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'GreenPartSegmentArrival', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [295] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(295, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 4L, 4L, 4L, 4L, 7L, 2L, 14L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(295, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 4999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 3, 1: 0, 2: 1, 3: 0, 4: 4, 5: 0, 6: 0, 7: 1, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 7L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-24 14:25:00,043 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-24 14:25:07,518 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-24 14:25:12,627 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1191) EOB(0) md5(6b98158cad4c21e5af65e568b0fd432e)

\*\*\* 2014-10-24 14:25:13,194 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1191) EOB(0) md5(43d6087f523a4e48481dcec2e33ab897)

\*\*\* 2014-10-24 14:25:13,610 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1191) EOB(0) md5(98fe31cdecc45f0d822e632198c97702)

\*\*\* 2014-10-24 14:25:14,629 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(1191) EOB(0) md5(8bac3f65a5ffa7d1c8b8a1052a9ceb14)

\*\*\* 2014-10-24 14:25:15,599 - pyltprecv - INFO - GreenPartReception.indication (16:1) len(236) EOB(1) md5(ba8c0f7de455764561622776b8883d9a)

\*\*\* 2014-10-24 14:25:15,599 - pyltprecv - INFO - Green Data Extents received for session (16, 1): (3574, 1191) (6191, 3809)

\*\*\* 2014-10-24 14:25:16,529 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 1) reason MISCOLORED

\*\*\* 2014-10-24 14:25:16,530 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 14:25:16,530 - pyltprecv - INFO - (0, 3574)

### Sender PCAP

14:25:06.981153 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:25:07.963990 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

14:25:08.946890 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

14:25:09.929699 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

14:25:10.912699 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(295)

14:25:11.132802 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(5000, 1191)

14:25:12.115449 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(6191, 1191)

14:25:13.098716 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(7382, 1191)

14:25:14.081727 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(8573, 1191)

14:25:15.064850 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x7 Green data, EOB C(6) Data(9764, 236)

14:25:15.976929 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(295)

14:25:17.035661 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(MISCOLORED)

14:25:17.039721 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xF Cancel-acknowledgment segment to block receiver

### Receiver PCAP

14:25:07.481256 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:25:08.464062 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

14:25:09.446959 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

14:25:12.615513 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(6191, 1191)

14:25:13.169578 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(3574, 1191)

14:25:13.598779 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(7382, 1191)

14:25:14.581804 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) x04 Green data, NOT EOB C(6) Data(8573, 1191)

14:25:15.564918 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x7 Green data, EOB C(6) Data(9764, 236)

14:25:16.476999 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4765, 235) CP(295)

14:25:16.535597 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(MISCOLORED)

14:25:17.539796 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xF Cancel-acknowledgment segment to block receiver

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-g file:greenPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/injectLTPTraffic.py -i br0 --skip 3 -T 0 --delay 2 -f miscolored

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6184), (6185, 7369), (7370, 8554), (8555, 9739), (9740, 9999)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 4999), (4741, 4999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 4999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4089 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['GreenPartSegmentArrival', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [4089] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4089, 0), (4089, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 2L, 4L, 4L, 4L, 4L, 7L, 2L, 14L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(4089, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 4999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 4, 1: 0, 2: 2, 3: 0, 4: 4, 5: 0, 6: 0, 7: 1, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 7L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 20 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414175359.736469 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 20)

\*\*\* 1414175359.736787 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414175359.738821 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414175369.917138 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 20)

\*\*\* 1414175371.254601 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 20) reason(3)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [(5000, 6184), (6185, 7369), (3556, 4740), (7370, 8554), (8555, 9739), (9740, 9999)] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (4741, 4999)] |
| AmountGreenDataReceivedByApp | 6185 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 4999 |
| GreenSegmentsReceived | [(5000, 6184), (6185, 7369), (3556, 4740), (7370, 8554), (8555, 9739), (9740, 9999)] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4089 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['GreenPartSegmentArrival', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [4089] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4089, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 4L, 4L, 4L, 4L, 4L, 7L, 2L, 14L, 15L] |
| cancelReasonCodes | [3] |
| checkpoints | [(4089, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | True |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 4999 |
| highestRedByteIndex | 4999 |
| lastCancelOrAck | 15 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 3, 1: 0, 2: 1, 3: 0, 4: 5, 5: 0, 6: 0, 7: 1, 8: 0, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 1, 15: 1} |
| segmentTypes | [0L, 2L, 4L, 7L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 20 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

ssssssssss[gs

### Sender PCAP

14:29:20.948935 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

14:29:22.155105 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

14:29:23.363666 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

14:29:24.571540 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

14:29:24.875587 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(4089)

14:29:26.082669 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(5000, 1185)

14:29:27.293646 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(6185, 1185)

14:29:28.502100 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(7370, 1185)

14:29:29.710901 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(8555, 1185)

14:29:29.912430 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x7 Green data, EOB C(6) Data(9740, 260)

14:29:30.215270 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(4089)

14:29:31.218805 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0xE Cancel segment from block receiver Reason(MISCOLORED)

14:29:31.259103 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0xF Cancel-acknowledgment segment to block receiver

### Receiver PCAP

14:29:21.448998 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

14:29:22.655169 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

14:29:23.863738 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

14:29:26.582727 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(5000, 1185)

14:29:27.793697 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(6185, 1185)

14:29:27.808876 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(3556, 1185)

14:29:29.002157 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(7370, 1185)

14:29:30.210974 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) x04 Green data, NOT EOB C(6) Data(8555, 1185)

14:29:30.412502 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x7 Green data, EOB C(6) Data(9740, 260)

14:29:30.715337 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0x2 Red data, Checkpoint, EORP, NOT EOB C(6) Data(4741, 259) CP(4089)

14:29:30.718730 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0xE Cancel segment from block receiver Reason(MISCOLORED)

14:29:31.759170 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 20) 0xF Cancel-acknowledgment segment to block receiver

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 2 , 4 , 7 , 14 , 15 | Good: Segment Types: 0 , 2 , 4 , 7 , 14 , 15 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 2 , 4 , 7 , 14 , 15 | Good: Segment Types: 0 , 2 , 4 , 7 , 14 , 15 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN Data | Good: CSID Offset Length CSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No Reports [] | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x03 | Good: CancelReasonCodes: 0x03 | Good: CancelReasonCodes: 0x03 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(295, 0), (295, 0)] | Good: Checkpoint transmitted: [(4089, 0), (4089, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(295, 2, False)] | Good: checkpoints retransmitted: [(4089, 2, False)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | Good: GreenPartReception.indication | Good: GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report sent. | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. | No check: both sides have transmission queues. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | Good: CancelACK to receiver | Good: CancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | Good: no cancel sent after cancel ack received. | Good: no cancel sent after cancel ack received. |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting data after cancel. | Good: Sender stops transmitting data after cancel. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (20) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (295) | Good: InitialCheckpointSerialNumber OK (4089) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 295 | Good: InitialCheckpointSerialNumber: 4089 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: SDALen

Service data aggregation(SDA) by length.

## Settings

### ION to Python

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 0 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

### Python to ION

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 0 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Aggregation service running at the sender and client; aggregation size at the sender set to 1,000 bytes, aggregation time set to 5 seconds.  
  
Sending client sends 'messages' to the aggregation service at the receiver. Each message is consists of 1 octet of message length followed by (length-1) bytes of 'message'. Sender chooses the lengths of the messages to send and ensures that it sends enough messages fast enough to invoke the 1,000-byte send trigger. All messages are sent as red-part data.

## Test-Specific Success Criteria

Receiver correctly demultiplexes the messages provided by the sending aggregator. Aggregate messages (LTP blocks) are transmitted according to the aggregation size limit setting.  
  
Verify that TransmissionSessionCompletion.indication indications are issued for each of the client SDUs.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONSDASender -d 18 -i node16.ionrc -l node16.ltprc\_aggregation

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -c 135 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

No automated checks for this test.

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16.ionrc

Processing option: l

Using ltprc file node16.ltprc\_aggregation

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

efgefgshgshg

#### ion\_sdasender.out

dddd

### Receiver Summary:

No automated checks for this test.

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-28 07:16:46,893 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 135

\*\*\* 2014-10-28 07:17:01,368 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(48) EOB(1) md5(df73d35e3bf38f20f0281b881b0a14ae)

\*\*\* 2014-10-28 07:17:01,368 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(48) EOB(1) md5(1052d12a187cb60879a9fd1405089644)

\*\*\* 2014-10-28 07:17:01,493 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 3) len(48) EOB(1) md5(5b1a298f379db8583fc86531574420f5)

\*\*\* 2014-10-28 07:17:01,494 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 3) len(48) EOB(1) md5(72f428de0661f430f35279f0a73d382d)

### Sender PCAP

07:17:00.837455 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 100) CP(1001)

07:17:00.947232 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 3) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 100) CP(5641)

07:17:01.878647 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1001, 62) [0 , 100]

07:17:01.882584 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(62)

07:17:02.001090 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 3) 0x8 Report segment RS(5641, 4000) [0 , 100]

07:17:02.004328 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 3) 0x9 Report-acknowledgment segment RA(4000)

### Receiver PCAP

07:17:01.337510 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 100) CP(1001)

07:17:01.378581 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1001, 62) [0 , 100]

07:17:01.447295 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 3) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 100) CP(5641)

07:17:01.501003 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 3) 0x8 Report segment RS(5641, 4000) [0 , 100]

07:17:02.382643 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(62)

07:17:02.504380 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 3) 0x9 Report-acknowledgment segment RA(4000)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -c 135 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:SDAInput.txt \

--maxAggregationLen 105 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONSDAReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 146)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 146 |
| EORP | 146 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4947 |
| InitialReportSerialNumber | 8106 |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4947, 0)] |
| allReports | [(8106, 4947)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(4947, 0)] |
| clientServiceID | 2 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 146 |
| highestRedByteIndex | 146 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 8106) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [8106] |
| reports | [(8106, 4947)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 100 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1415035616.030546 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1415035616.030702 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1415035618.039904 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 100)

\*\*\* 1415035618.040193 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 100)

\*\*\* 1415035618.040426 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 100)

\*\*\* 1415035618.245857 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 100)

\*\*\* 1415035618.246008 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 100)

\*\*\* 1415035618.246285 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 100)

\*\*\* 1415035619.301631 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 100)

\*\*\* 1415035619.301854 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 100)

\*\*\* 1415035619.302068 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 100)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 146)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 146 |
| EORP | 146 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4947 |
| InitialReportSerialNumber | 8106 |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4947, 0)] |
| allReports | [(8106, 4947)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(4947, 0)] |
| clientServiceID | 2 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 146 |
| highestRedByteIndex | 146 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 8106) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [8106] |
| reports | [(8106, 4947)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 100 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

stgs

#### ion\_sdareceiver.out

handleItem for clientID 135 gets: 0\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 1\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 2\_Please note that this is an sdatest string.

### Sender PCAP

12:26:58.245563 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 100) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 147) CP(4947)

12:26:59.248549 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 100) 0x8 Report segment RS(4947, 8106) [0 , 147]

12:26:59.300153 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 100) 0x9 Report-acknowledgment segment RA(8106)

### Receiver PCAP

12:26:58.745618 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 100) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 147) CP(4947)

12:26:58.748476 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 100) 0x8 Report segment RS(4947, 8106) [0 , 147]

12:26:59.800209 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 100) 0x9 Report-acknowledgment segment RA(8106)

## Comments:

### General Comments

Note: Configuration information (specifically the RED\_SIZE) is ignored for SDA tests. Other mechanisms are used to exercise the SDA interface.

### ION to Python Comments

All of these checks done by hand.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types 3, 8, 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types 3, 8, 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN | Good: RSN RSN\_CSN [(8106, 4947)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement sent | Good: Report Acknowledgement(s) sent: [8106] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | No checkpoints retransmitted | Good: Checkpoint transmitted: [(4947, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No checkpoints retransmitted | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports retransmitted | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No checkpoints retransmitted | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: multiple client ADUs received. | Good: received 4 client ADUs. | Good: received 3 client ADUs |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: reception reports sent. | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No test. | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | No Check | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segment sent. | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segment sent. | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No cancel segment sent. | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment sent. | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancel segment sent. | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | No cancel segment sent. | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancel segment sent. | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segment sent. | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segment sent. | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored segments. | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | No: Not required by SDA | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | No: Not required by SDA | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | No | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | No | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | See SDA: InitialTransmissionCompletion.indication below | See SDA: InitialTransmissionCompletion.indication below |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No: Not required by SDA | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  | Ensure: multiple client ADUs received. | Good: received 4 client ADUs. | Good: received 3 client ADUs |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  | Check for this | Good: SDA data was sent. | Good: SDA data was sent. |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No test -- implementation dependent. | No test -- implementation dependent. |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  | Check for this | Good: Verified by hand. | Good: Verified by hand. |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  | Check for this | Good: Only Red-Part data seen on the wire. | Good: Only Red-Part data seen on the wire. |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  | Check for this | Good: Verified by hand. | Good: Verified by hand. |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  | No check: using aggregation by size. | No | No |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  | Look for transmissionsessioncompletion.indicatins. | No check | No check |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  | Ensure: multiple client ADUs received. | Good: 4 client ADUs blocks received. | Good: 3 client ADUs blocks received. |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No check | No check |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1, 3) | Good: SessionNumber OK (100) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: Verified by hand. | Good: InitialCheckpointSerialNumber OK (4947) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 1001, 5641 | Good: InitialCheckpointSerialNumber: 4947 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No unknown LTP extensions. | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: SDATime

Service data aggregation (SDA) by time.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 10 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Aggregation service running at the sender and client; aggregation size at the sender set to 1,000 bytes, aggregation time set to 5 seconds.  
  
Sending client uses the aggregation service to send a few messages, staying under the 1,000-byte aggregation size trigger. Verify that aggregated groups are send out approx. every 5s.

## Test-Specific Success Criteria

Receiver correctly demultiplexes the messages provided by the sending aggregator. Aggregate messages (LTP blocks) are transmitted according to the aggregation time setting.  
  
  
Verify that TransmissionSessionCompletion.indication indications are issued for each of the client SDUs.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONSDASender -d 18 -i node16.ionrc -l node16.ltprc\_aggregation\_time

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 --enableSDA True -c 135 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 199)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 199 |
| EORP | 199 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3961 |
| InitialReportSerialNumber | 13673 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(3961, 0)] |
| allReports | [(13673, 3961)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(3961, 0)] |
| clientServiceID | 2 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 199 |
| highestRedByteIndex | 199 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 13673) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [13673] |
| reports | [(13673, 3961)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16.ionrc

Processing option: l

Using ltprc file node16.ltprc\_aggregation\_time

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

efgshg

#### ion\_sdasender.out

dddd

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 199)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 199 |
| EORP | 199 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3961 |
| InitialReportSerialNumber | 13673 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(3961, 0)] |
| allReports | [(13673, 3961)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(3961, 0)] |
| clientServiceID | 2 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 199 |
| highestRedByteIndex | 199 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 13673) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [13673] |
| reports | [(13673, 3961)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-11-03 11:44:00,835 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 135

\*\*\* 2014-11-03 11:44:35,263 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(48) EOB(1) md5(df73d35e3bf38f20f0281b881b0a14ae)

\*\*\* 2014-11-03 11:44:35,264 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(48) EOB(1) md5(1052d12a187cb60879a9fd1405089644)

\*\*\* 2014-11-03 11:44:35,264 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(48) EOB(1) md5(5b1a298f379db8583fc86531574420f5)

\*\*\* 2014-11-03 11:44:35,264 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(48) EOB(1) md5(72f428de0661f430f35279f0a73d382d)

### Sender PCAP

11:44:34.760644 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 200) CP(3961)

11:44:35.769985 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(3961, 13673) [0 , 200]

11:44:35.771810 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(13673)

### Receiver PCAP

11:44:35.260741 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 200) CP(3961)

11:44:35.269932 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(3961, 13673) [0 , 200]

11:44:36.271869 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(13673)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -c 135 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:SDAInput.txt \

--maxAggregationTime 15 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONSDAReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 739)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 739 |
| EORP | 739 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 821 |
| InitialReportSerialNumber | 1805 |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(821, 0)] |
| allReports | [(1805, 821)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(821, 0)] |
| clientServiceID | 2 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 739 |
| highestRedByteIndex | 739 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 1805) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [1805] |
| reports | [(1805, 821)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 78 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414522310.750286 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414522310.750596 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414522325.773513 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.773825 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.774000 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.774158 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.774394 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.774558 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.774819 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.774988 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.775145 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.775302 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.775468 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.775624 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.775816 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.775976 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522325.776132 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414522326.588456 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.589346 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.589958 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.590396 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.590806 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.591325 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.591798 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.592235 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.592626 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.593166 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.593550 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.594109 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.594601 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.595059 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522326.595429 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414522327.629916 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.630327 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.630494 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.630778 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.630964 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.631222 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.631595 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.631722 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.631922 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.632184 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.632617 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.632741 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.632945 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.633128 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

\*\*\* 1414522327.633388 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 78)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 739)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 739 |
| EORP | 739 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 821 |
| InitialReportSerialNumber | 1805 |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(821, 0)] |
| allReports | [(1805, 821)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(821, 0)] |
| clientServiceID | 2 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 739 |
| highestRedByteIndex | 739 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 1805) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [1805] |
| reports | [(1805, 821)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 78 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

stgs

#### ion\_sdareceiver.out

handleItem for clientID 135 gets: 0\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 1\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 2\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 3\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 4\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 5\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 6\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 7\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 8\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 9\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 10\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 11\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 12\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 13\_Please note that this is an sdatest string.

handleItem for clientID 135 gets: 14\_Please note that this is an sdatest string.

### Sender PCAP

14:52:06.584978 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 740) CP(821)

14:52:07.589095 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment RS(821, 1805) [0 , 740]

14:52:07.626614 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x9 Report-acknowledgment segment RA(1805)

### Receiver PCAP

14:52:07.085031 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(0, 740) CP(821)

14:52:07.089041 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment RS(821, 1805) [0 , 740]

14:52:08.126661 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x9 Report-acknowledgment segment RA(1805)

## Comments:

### General Comments

Note: Configuration information (specifically the RED\_SIZE) is ignored for SDA tests. Other mechanisms are used to exercise the SDA interface.

### Python to ION Comments

Automated check for ReceiveIndications for Python->ION check would fail. Checked by hand.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(13673, 3961)] | Good: RSN RSN\_CSN [(1805, 821)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [13673] | Good: Report Acknowledgement(s) sent: [1805] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(3961, 0)] | Good: Checkpoint transmitted: [(821, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: multiple client ADUs received. | Good: received 4 client ADUs. | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | See SDA: InitialTransmissionCompletion.indication below |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | NO ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  | Ensure: multiple client ADUs received. | Good: received 4 client ADUs. | Good: received 14 client ADUs. |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  | Check for this | Good: SDA data was sent. | Good: SDA data was sent. |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No test -- implementation dependent. | No test -- implementation dependent. |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  | Check for this | Good: Verified by hand. | Good: Verified by hand. |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  | Check for this | Good: Verified by hand. | Good: Verified by hand. |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  | No check: using aggregation by time. | No check: using aggregation by time. | No check: using aggregation by time. |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  | Check for this | Good: Verified by hand. | Good: Verified by hand. |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  | Look for transmissionsessioncompletion.indicatins. | Good: InitialTransmissionCompletion.indications provided. | Good: InitialTransmissionCompletion.indications provided. |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  | Ensure: multiple client ADUs received. | Good: received 4 client ADUs. | Good: received 14 client ADUs. |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No check | No check |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (78) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (3961) | Good: InitialCheckpointSerialNumber OK (821) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 3961 | Good: InitialCheckpointSerialNumber: 821 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: SDACancel

Cancel SDA receive session and ensure that N ReceptionSessionCancellation.indications are sent to the LTP client (1 per SDA client capsule).

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 5000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Cancel an SDA session from the receiver and ensure that N (where N is the number of data capsules in the SDA aggregate) ReceptionSessionCancellation.indications to the clients.

## Test-Specific Success Criteria

Ensure that N ReceptionSessionCancellation.indications are sent to the client(s). Note that the value of N depends not only on the amount of data received before cancellation, but also on the client services used (since those determine how to extract capsules from the overall block).

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONSDASender -d 18 -i node16.ionrc -m 1000 -n 12 -l node16.ltprc\_aggregation\_large

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -c 135 --enableSDA True -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, }}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -T 0 -S 6 -R 12 -s 10.0.0.1 -s 10.0.0.2 -t 500

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146), (7147, 8337), (8338, 9528), (9529, 10462), (9529, 10462), (9529, 10462), (9529, 10462), (9529, 10462), (9529, 10462), (9529, 10462), (9529, 10462), (9529, 10462), (9529, 10462)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 10462 |
| EORP | 10462 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 7098 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [7098] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 13L] |
| cancelReasonCodes | [2, 2] |
| checkpoints | [(7098, 0)] |
| clientServiceID | 2 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 10462 |
| highestRedByteIndex | 10462 |
| lastCancelOrAck | 13 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 2, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16.ionrc

Processing option: l

Using ltprc file node16.ltprc\_aggregation\_large

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeefggggggggg=g=g=g=g=g=g=g=g=g{ggs

#### ion\_sdasender.out

ddddddddddd

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | None |
| EORP | None |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | False |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 12L, 13L] |
| cancelReasonCodes | [2] |
| checkpoints | [] |
| clientServiceID | 2 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | False |
| highestRedByteIndex | 7146 |
| lastCancelOrAck | 13 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 6, 1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [0L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-30 10:01:12,321 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 135

\*\*\* 2014-10-30 10:03:23,734 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-30 10:03:23,738 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-30 10:03:23,741 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-30 10:03:23,744 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-30 10:03:23,746 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-30 10:03:23,748 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-30 10:03:23,749 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

### Sender PCAP

10:01:26.222819 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(0, 1192)

10:01:33.240020 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(1192, 1191)

10:01:40.257493 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(2383, 1191)

10:01:47.274950 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(3574, 1191)

10:01:54.292431 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(4765, 1191)

10:02:01.310350 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(5956, 1191)

10:02:08.327878 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(7147, 1191)

10:02:15.345362 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(8338, 1191)

10:02:22.363068 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:02:27.929133 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:02:33.495162 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:02:39.061372 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:02:44.627345 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:02:50.193460 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:02:55.759540 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:03:01.325516 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:03:06.891618 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:03:12.457731 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(2) Data(9529, 934) CP(7098)

10:03:18.023874 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:03:23.200402 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:03:24.251079 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender

### Receiver PCAP

10:01:26.722850 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(0, 1192)

10:01:33.740108 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(1192, 1191)

10:01:40.757550 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(2383, 1191)

10:01:47.775015 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(3574, 1191)

10:01:54.792494 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(4765, 1191)

10:02:01.810421 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(2) Data(5956, 1191)

10:03:23.700465 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:03:23.751023 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender

## There Are No Python to ION Results:

There are no results for this test.

## Comments:

### Python to ION Comments

Not implemented by ION LTP.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 12 , 13 |  |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 12 , 13 |  |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions |  |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions |  |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN Data |  |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No Reports [] |  |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements |  |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes : 0x02 | Good: CancelReasonCodes: 0x02 |  |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers |  |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request |  |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | Good: verified by hand |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. |  |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission |  |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0), (7098, 0)] |  |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. |  |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports transmitted |  |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted |  |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(7098, 10, False)] |  |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. |  |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No reports transmitted |  |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted |  |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No RedPartReception.indication |  |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication |  |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent |  |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent |  |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication |  |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission |  |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report sent. |  |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions |  |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | Good: CancelACK to sender |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver |  |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | Good: no cancel sent after cancel ack received. |  |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting after cancel. |  |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession |  |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present |  |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication |  |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | No RedPartReception.indication |  |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication |  |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication |  |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication |  |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication |  |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | NO ReceptionSessionStart.indication |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  | Ensure: No ADUs received | Good: no ADUs received |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: SDA data was sent. |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No test -- implementation dependent. |  |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | Good: Verified by hand. |  |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | Good: Only Red-Part data seen on the wire. |  |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No check |  |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No check |  |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No check |  |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No check |  |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  | Ensure: Check for this | Good: multiple ReceptionCancellation.indications delivered to app. |  |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram |  |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram |  |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA |  |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 |  |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth |  |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite |  |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check |  |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check |  |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check |  |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. |  |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure |  |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. |  |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception |  |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) |  |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (7098) |  |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 7098 |  |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found |  |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet |  |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet |  |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP |  |

# Test: Auth1

10KB Red data, ciphersuite 0 both directions; good keys.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Send a red block with LTP Authentication ciphersuite 0 turned on in both directions. Authenticate all segments (data, reports, acks).

## Test-Specific Success Criteria

Ensure that authentication is invoked and that the block is received correctly.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -S node16.ionsecrc\_0\_11\_22 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 0, 'xmitKeyID': '50', 'xmitKeyMaterial': '3232'}], 'recvAuthMaterial': [{'recvCiphersuite': 0, 'recvKeyID': '23', 'recvKeyMaterial': '3131'}]}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 221 |
| InitialReportSerialNumber | 4196 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(221, 0)] |
| allReports | [(4196, 221)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(221, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 4196) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [4196] |
| reports | [(4196, 221)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_0\_11\_22

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_0\_11\_22

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 16

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '16' ciphersuite '0' key name '11' frequency '0'

Transmission rule: ltp engine id '18' ciphersuite '0' key name '11' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '0' key name '22' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '0' key name '22' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

efgshg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 221 |
| InitialReportSerialNumber | 4196 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(221, 0)] |
| allReports | [(4196, 221)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(221, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 4196) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [4196] |
| reports | [(4196, 221)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-27 13:19:38,804 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-27 13:19:48,279 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-10-27 13:19:48,280 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 2) len(100) EOB(1) md5(7acedd1a84a4cfcb6e7a16003242945e)

### Sender PCAP

13:19:47.730376 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(221) TExt1 (T=0, L=10)

13:19:48.788062 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(221, 4196) [0 , 100] TExt1 (T=0, L=10)

13:19:48.792973 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment HExt1 (T=0, L=2) RA(4196) TExt1 (T=0, L=10)

### Receiver PCAP

13:19:48.230439 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(221) TExt1 (T=0, L=10)

13:19:48.288003 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(221, 4196) [0 , 100] TExt1 (T=0, L=10)

13:19:49.293050 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment HExt1 (T=0, L=2) RA(4196) TExt1 (T=0, L=10)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 0, 'xmitKeyID': '50', 'xmitKeyMaterial': '3232'}], 'recvAuthMaterial': [{'recvCiphersuite': 0, 'recvKeyID': '23', 'recvKeyMaterial': '3131'}]}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -S node16.ionsecrc\_0\_11\_22

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 11423 |
| InitialReportSerialNumber | 577 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(11423, 0)] |
| allReports | [(577, 11423)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(11423, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 577) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [577] |
| reports | [(577, 11423)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 12 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414150578.084890 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 12)

\*\*\* 1414150578.085278 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414150578.088069 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414150578.292090 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 12)

\*\*\* 1414150579.354164 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 12)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 11423 |
| InitialReportSerialNumber | 577 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(11423, 0)] |
| allReports | [(577, 11423)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(11423, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 577) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [577] |
| reports | [(577, 11423)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 12 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_0\_11\_22

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_0\_11\_22

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 16

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '16' ciphersuite '0' key name '11' frequency '0'

Transmission rule: ltp engine id '18' ciphersuite '0' key name '11' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '0' key name '22' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '0' key name '22' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

stgs

### Sender PCAP

07:36:18.293276 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 12) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(11423) TExt1 (T=0, L=10)

07:36:19.298516 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 12) 0x8 Report segment HExt1 (T=0, L=2) RS(11423, 577) [0 , 100] TExt1 (T=0, L=10)

07:36:19.355158 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 12) 0x9 Report-acknowledgment segment HExt1 (T=0, L=2) RA(577) TExt1 (T=0, L=10)

### Receiver PCAP

07:36:18.793334 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 12) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(11423) TExt1 (T=0, L=10)

07:36:18.798465 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 12) 0x8 Report segment HExt1 (T=0, L=2) RS(11423, 577) [0 , 100] TExt1 (T=0, L=10)

07:36:19.855201 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 12) 0x9 Report-acknowledgment segment HExt1 (T=0, L=2) RA(577) TExt1 (T=0, L=10)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  | Ensure: Header / Trailer Extensions | Good: Header / Trailer Extensions | Good: Header / Trailer Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  | Ensure: Authentication uses headers. | Good: Authentication uses headers. | Good: Authentication uses headers. |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(4196, 221)] | Good: RSN RSN\_CSN [(577, 11423)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [4196] | Good: Report Acknowledgement(s) sent: [577] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: No cancel segments sent | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  | Ensure: Trailers | Good: Authentication uses trailers. | Good: Authentication uses trailers. |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  | Ensure: TransmissionRequest.request | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(221, 0)] | Good: Checkpoint transmitted: [(11423, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  | Ensure: Report transmitted | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  | Ensure: SendReceptionReport | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | Good: 1 | Good: 1 |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  | Ensure: Auth RFC5327 | Good: Auth RFC5327 | Good: Auth RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  | Ensure: Auth May use authentication | Good: Auth May use authentication | Good: Auth May use authentication |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  | Ensure this | Good: verified by examination of pcap file. | Good: verified by examination of pcap file. |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  | Ensure: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  | Ensure this | Good: verified by examination of configuration files. | Good: verified by examination of configuration files. |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  | Can't check this; good auth material | No check:Good authentication material. | No check:Good authentication material. |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  | Ensure: AUTH -- all or none | Good: AUTH -- all or none | Good: AUTH -- all or none |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  | Ensure: AUTH -- extension structure | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (12) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (221) | Good: InitialCheckpointSerialNumber OK (11423) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 221 | Good: InitialCheckpointSerialNumber: 11423 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Auth1b

10KB Red data, ciphersuite 0 both directions; bad sender to receiver key.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Send a red block with LTP Authentication ciphersuite 0 turned on in both directions. Authenticate all segments (data, reports, acks) but using the WRONG KEYS for the sender -> receiver direction.

## Test-Specific Success Criteria

Data segments rejected by receiver.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -S node16.ionsecrc\_0\_11\_22 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 0, 'xmitKeyID': '50', 'xmitKeyMaterial': '3333'}], 'recvAuthMaterial': [{'recvCiphersuite': 0, 'recvKeyID': '99', 'recvKeyMaterial': '3535'}]}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4324 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [4324] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(4324, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 10, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_0\_11\_22

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_0\_11\_22

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 16

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '16' ciphersuite '0' key name '11' frequency '0'

Transmission rule: ltp engine id '18' ciphersuite '0' key name '11' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '0' key name '22' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '0' key name '22' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

efg=g=g=g=g=g=g=g=g=g{gggggggggg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4324 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [4324] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(4324, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 10, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-31 08:43:26,571 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

### Sender PCAP

08:43:35.801665 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:43:40.798125 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:43:45.799571 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:43:50.800649 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:43:55.801467 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:00.803105 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:05.804049 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:10.804672 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:15.806335 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:20.807518 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:25.809026 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:30.811186 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:35.812725 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:40.813536 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:45.815300 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:50.816884 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:55.817841 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:45:00.819307 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:45:05.820943 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:45:10.821528 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

### Receiver PCAP

08:43:36.301728 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:43:41.298214 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:43:46.299679 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:43:51.300726 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:43:56.301522 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:01.303179 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:06.304132 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:11.304713 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:16.306404 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:21.307593 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4324) TExt1 (T=0, L=10)

08:44:26.309092 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:31.311228 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:36.312777 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:41.313616 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:46.315376 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:51.316946 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:44:56.317902 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:45:01.319355 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:45:06.321022 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

08:45:11.321598 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 0, 'xmitKeyID': '25', 'xmitKeyMaterial': '3535'}], 'recvAuthMaterial': [{'recvCiphersuite': 0, 'recvKeyID': '23', 'recvKeyMaterial': '3232'}]}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -S node16.ionsecrc\_0\_11\_22

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4477 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [4477] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L] |
| cancelReasonCodes | [2] |
| checkpoints | [(4477, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 41 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1415035937.608545 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 41)

\*\*\* 1415035937.608831 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1415035937.610515 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1415035937.814217 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 41)

\*\*\* 1415035979.864580 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 41) reason(2)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4477 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [4477] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0), (4477, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L] |
| cancelReasonCodes | [2] |
| checkpoints | [(4477, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 41 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_0\_11\_22

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_0\_11\_22

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 16

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '16' ciphersuite '0' key name '11' frequency '0'

Transmission rule: ltp engine id '18' ciphersuite '0' key name '11' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '0' key name '22' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '0' key name '22' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

ssssssssssssssssssssss

### Sender PCAP

12:32:17.815119 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:19.815162 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:21.817727 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:23.823793 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:25.823523 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:27.829025 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:29.829888 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:31.832058 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:33.834719 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:35.837589 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:37.839852 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:39.842709 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:41.844959 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:43.846929 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:45.849515 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:47.852576 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:49.854871 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:51.856956 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:53.859369 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:55.861154 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:57.863097 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:59.869271 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

### Receiver PCAP

12:32:18.315165 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:20.315236 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:22.317768 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:24.323888 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:26.323601 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:28.329065 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:30.329953 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:32.332123 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:34.334753 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:36.337629 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:38.339927 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:40.342746 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:42.345011 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:44.347013 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:46.349558 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:48.352662 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:50.354912 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:52.357009 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:54.359420 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:56.361219 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:32:58.363137 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4477) TExt1 (T=0, L=10)

12:33:00.369335 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 41) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 12 | Good: Segment Types: 3 , 12 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) | Ensure: Segment Types: 3 | Good: Segment Types: 3 , 12 | Good: Segment Types: 3 , 12 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  | Ensure: Header / Trailer Extensions | Good: Header / Trailer Extensions | Good: Header / Trailer Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  | Ensure: Authentication uses headers. | Good: Authentication uses headers. | Good: Authentication uses headers. |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN Data | Good: CSID Offset Length CSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No Reports [] | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes : 0x02 | Good: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  | Ensure: Trailers | Good: Authentication uses trailers. | Good: Authentication uses trailers. |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  | Ensure: TransmissionRequest.request | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0), (4324, 0)] | Good: Checkpoint transmitted: [(8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0), (8109, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  | Ensure: No reports transmitted | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  | Ensure: No reports transmitted | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(4324, 10, False)] | Good: checkpoints retransmitted: [(8109, 21, False)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  | Ensure: No reports transmitted | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: No RedPartReception.indication | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  | Ensure: No GreenPartReception.indication | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report sent. | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | Good: Session cancelled after too many Cancel retransmissions by sender. | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. | No check: both sides have transmission queues. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel ack received (was looking for 13 in response to 12 in [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L]) | No cancel ack received (was looking for 13 in response to 12 in [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L]) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting data after cancel. | Good: Sender stops transmitting data after cancel. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: No RedPartReception.indication | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: No TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | NO ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | Good: 1 | Good: 1 |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  | Ensure: Auth RFC5327 | Good: Auth RFC5327 | Good: Auth RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  | Ensure: Auth May use authentication | Good: Auth May use authentication | Good: Auth May use authentication |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  | Ensure this | Good: verified by examination of pcap file. | Good: verified by examination of pcap file. |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  | Ensure: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  | Ensure this | Good: verified by examination of configuration files. | Good: verified by examination of configuration files. |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  | Ensure: segments with bad authentication rejected. | Good: segments with bad authentication rejected. | Good: segments with bad authentication rejected. |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  | Ensure: AUTH -- all or none | Good: AUTH -- all or none | Good: AUTH -- all or none |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  | Ensure: AUTH -- extension structure | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (71) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (4324) | Good: InitialCheckpointSerialNumber OK (8109) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 4324 | Good: InitialCheckpointSerialNumber: 8109 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Auth1c

10KB Red data, ciphersuite 0 both directions; bad receiver to sender key.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Send a red block with LTP Authentication ciphersuite 0 turned on in both directions. Authenticate all segments (data, reports, acks) but using the WRONG KEYS for the receiver -> sender direction.

## Test-Specific Success Criteria

Segments accepted by receiver; report segments rejected by the sender.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -S node16.ionsecrc\_0\_11\_22 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 0, 'xmitKeyID': '50', 'xmitKeyMaterial': '3535'}], 'recvAuthMaterial': [{'recvCiphersuite': 0, 'recvKeyID': '99', 'recvKeyMaterial': '3131'}]}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [3L] |
| InitialCheckpointSerialNumber | 4869 |
| InitialReportSerialNumber | 1500 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [1500] |
| allCheckpoints | [(4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0)] |
| allReports | [(1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869)] |
| allSegmentsInOrder | [3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 14L, 3L, 14L, 14L, 12L, 14L, 13L, 14L, 14L, 12L, 13L, 14L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(4869, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 14 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | True |
| reports | [(1500, 4869)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 22, 9: 0, 10: 0, 11: 0, 12: 2, 13: 2, 14: 7, 15: 0} |
| segmentTypes | [3L, 8L, 12L, 13L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_0\_11\_22

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_0\_11\_22

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 16

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '16' ciphersuite '0' key name '11' frequency '0'

Transmission rule: ltp engine id '18' ciphersuite '0' key name '11' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '0' key name '22' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '0' key name '22' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

efgs

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [3L] |
| InitialCheckpointSerialNumber | 4869 |
| InitialReportSerialNumber | 1500 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [1500] |
| allCheckpoints | [(4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0)] |
| allReports | [(1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869)] |
| allSegmentsInOrder | [3L, 8L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 14L, 3L, 14L, 14L, 14L, 12L, 13L, 14L, 14L, 12L, 13L, 14L, 14L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(4869, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 14 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | True |
| reports | [(1500, 4869)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 22, 9: 0, 10: 0, 11: 0, 12: 2, 13: 2, 14: 8, 15: 0} |
| segmentTypes | [3L, 8L, 12L, 13L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-24 07:46:22,313 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-24 07:46:31,790 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-10-24 07:46:31,792 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 2) len(100) EOB(1) md5(7acedd1a84a4cfcb6e7a16003242945e)

\*\*\* 2014-10-24 07:47:15,882 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-24 07:47:15,882 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 07:47:15,883 - pyltprecv - INFO - (0, 100)

\*\*\* 2014-10-24 07:47:21,955 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-24 07:47:21,955 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 07:47:21,955 - pyltprecv - INFO -

\*\*\* 2014-10-24 07:47:26,976 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-24 07:47:26,977 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 07:47:26,977 - pyltprecv - INFO -

\*\*\* 2014-10-24 07:47:31,956 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-24 07:47:31,957 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 07:47:31,958 - pyltprecv - INFO -

### Sender PCAP

07:46:31.271207 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:32.302551 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:34.306184 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:36.307998 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:36.418658 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:38.311887 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:40.314127 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:41.419984 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:42.314878 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:44.318048 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:46.319392 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:46.420962 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:48.324013 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:50.332315 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:51.421808 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:52.335676 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:54.342274 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:56.345366 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:56.422521 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:58.349653 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:00.353985 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:01.423350 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:47:02.356466 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:04.361170 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:06.365550 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:06.424654 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:47:08.368622 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:10.372844 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:11.425451 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:47:12.375197 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:14.377562 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:16.388605 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:16.426327 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:47:18.394124 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:20.398505 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:21.427942 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:22.402713 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:22.459337 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:24.407212 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:26.409358 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:26.428565 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:27.480086 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:28.412964 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

### Receiver PCAP

07:46:31.771267 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:31.802428 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:33.806133 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:35.807917 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:36.918723 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:37.811816 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:39.814060 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:41.814785 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:41.920092 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:43.817994 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:45.819310 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:46.921027 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:47.823965 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:49.832214 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:51.835632 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:51.921880 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:53.842180 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:55.845299 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:56.922590 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:46:57.849533 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:46:59.853915 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:01.856406 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:01.923418 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:47:03.861104 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:05.865489 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:06.924724 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:47:07.868541 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:09.872711 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:11.875138 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:11.925518 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:47:13.877510 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(4869, 1500) [0 , 100] TExt1 (T=0, L=10)

07:47:15.888545 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:16.926390 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4869) TExt1 (T=0, L=10)

07:47:17.894062 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:19.898453 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:21.902580 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:21.928008 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:21.959286 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:23.907139 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:25.909300 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:26.928670 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:26.980012 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:27.912883 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:47:29.914241 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 0, 'xmitKeyID': '25', 'xmitKeyMaterial': '3232'}], 'recvAuthMaterial': [{'recvCiphersuite': 0, 'recvKeyID': '23', 'recvKeyMaterial': '3535'}]}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -S node16.ionsecrc\_0\_11\_22

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3925 |
| InitialReportSerialNumber | 7571 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [7571] |
| allCheckpoints | [(3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0)] |
| allReports | [(7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925)] |
| allSegmentsInOrder | [3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 12L, 13L] |
| cancelReasonCodes | [2] |
| checkpoints | [(3925, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 13 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [(7571, 3925)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 9, 9: 0, 10: 0, 11: 0, 12: 1, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 78 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414151733.852252 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 78)

\*\*\* 1414151733.852397 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414151733.853996 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414151734.056346 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 78)

\*\*\* 1414151776.118458 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 78) reason(2)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3925 |
| InitialReportSerialNumber | 7571 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [7571] |
| allCheckpoints | [(3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0)] |
| allReports | [(7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925)] |
| allSegmentsInOrder | [3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 12L, 13L] |
| cancelReasonCodes | [2] |
| checkpoints | [(3925, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 13 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [(7571, 3925)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 9, 9: 0, 10: 0, 11: 0, 12: 1, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 78 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_0\_11\_22

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_0\_11\_22

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 16

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: 11

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '16' ciphersuite '0' key name '11' frequency '0'

Transmission rule: ltp engine id '18' ciphersuite '0' key name '11' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '0' key name '22' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '0' key name '22' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

stgss+gsss+gss+gsss+gss+gsss+gss+gsss+gs}g

### Sender PCAP

07:55:34.056984 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:35.060004 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:36.061224 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:38.063484 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:40.065398 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:40.258841 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:42.067002 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:44.069219 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:45.259702 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:46.072791 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:48.075968 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:50.080168 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:50.261091 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:52.081391 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:54.082007 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:55.261733 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:56.085566 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:58.090420 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:00.090771 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:00.262625 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:56:02.093928 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:04.098226 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:05.263760 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:56:06.100960 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:08.102736 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:10.105637 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:10.264659 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:56:12.111903 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:14.119596 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:15.265542 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:56:16.125606 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:56:17.127691 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=10)

### Receiver PCAP

07:55:34.557040 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:34.559963 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:36.561279 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:38.563549 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:39.758753 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:40.565450 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:42.567095 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:44.569284 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:44.759634 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:46.572834 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:48.576041 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:49.761042 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:50.580296 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:52.581463 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:54.582084 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:54.761677 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:55:56.585619 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:58.590468 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:55:59.762527 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:56:00.590864 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:02.594014 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:04.598300 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:04.763709 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:56:06.601023 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:08.602816 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:09.764561 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:56:10.605699 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:12.611979 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:14.619709 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3925) TExt1 (T=0, L=10)

07:56:14.765499 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0x8 Report segment HExt1 (T=0, L=2) RS(3925, 7571) [0 , 100] TExt1 (T=0, L=10)

07:56:16.625690 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=10)

07:56:16.627640 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 78) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=10)

## Comments:

### General Comments

Red-Part reception will work here; the receiving LTP engine gets the data and delivers it to the application. The receiving LTP engine just can't (successfully) acknowledge the data.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 12 , 13 , 14 | Good: Segment Types: 3 , 8 , 12 , 13 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 8 , 12 , 13 , 14 | Good: Segment Types: 3 , 8 , 12 , 13 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  | Ensure: Header / Trailer Extensions | Good: Header / Trailer Extensions | Good: Header / Trailer Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  | Ensure: Authentication uses headers. | Good: Authentication uses headers. | Good: Authentication uses headers. |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN Data | Good: CSID Offset Length CSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869), (1500, 4869)] | Good: RSN RSN\_CSN [(7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925), (7571, 3925)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes : 0x02 | Good: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  | Ensure: Trailers | Good: Authentication uses trailers. | Good: Authentication uses trailers. |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  | Ensure: TransmissionRequest.request | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0), (4869, 0)] | Good: Checkpoint transmitted: [(3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0), (3925, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  | Ensure: Reports retransmitted | Good: StartReportSegment Timer (Report(s) retransmitted) | Good: StartReportSegment Timer (Report(s) retransmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(4869, 10, 1500)] | Good: checkpoints retransmitted: [(3925, 21, 7571)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  | Ensure: Report transmitted | Good: Report retransmitted | Good: Report retransmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No check: sender cancels first. | No check: sender cancels first. |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: RedpartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  | Ensure: SendReceptionReport | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  | Ensure: no report ack | No report ack | No report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | Good: Session cancelled after too many Cancel retransmissions by sender. | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  | Ensure that this happens. | Good: Cancel segment retransmission | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. | No check: both sides have transmission queues. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | Good: CancelACK to sender | Good: CancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No check: only the receiver cancels and the sender can't ack it. | Good: no cancel sent after cancel ack received. |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No check: sender can't receive (authenticate) cancel segment. | No check: sender can't receive (authenticate) cancel segment. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: No TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionStart.indication | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | Good: 1 | Good: 1 |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  | Ensure: Auth RFC5327 | Good: Auth RFC5327 | Good: Auth RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  | Ensure: Auth May use authentication | Good: Auth May use authentication | Good: Auth May use authentication |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  | Ensure this | Good: verified by examination of configuration files. | Good: verified by examination of pcap file. |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  | Ensure: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  | Ensure this | Good: AUTH -- extension structure | Good: verified by examination of configuration files. |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  | Ensure: segments with bad authentication rejected. | Good: segments with bad authentication rejected. | Good: segments with bad authentication rejected. |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  | Ensure: AUTH -- all or none | Good: AUTH -- all or none | Good: AUTH -- all or none |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  | Ensure: AUTH -- extension structure | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (78) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (4869) | Good: InitialCheckpointSerialNumber OK (3925) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 4869 | Good: InitialCheckpointSerialNumber: 3925 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Auth2

10KB Red data, ciphersuite 1 both directions; good keys.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Like Auth1 but with ciphersuite 1

## Test-Specific Success Criteria

Ensure that authentication is invoked and that the block is received correctly.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -S node16.ionsecrc\_1\_tkey1\_rkey2 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 1, 'xmitKeyID': '"01"', 'xmitKeyMaterial': 'key2\_priv.der'}], 'recvAuthMaterial': [{'recvCiphersuite': 1, 'recvKeyID': '"01"', 'recvKeyMaterial': 'key1\_pub.der'}]}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4045 |
| InitialReportSerialNumber | 11919 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4045, 0)] |
| allReports | [(11919, 4045)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(4045, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 11919) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [11919] |
| reports | [(11919, 4045)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_1\_tkey1\_rkey2

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_1\_tkey1\_rkey2

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: key3

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '18' ciphersuite '1' key name 'key3' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '1' key name 'key2' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '1' key name 'key1' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

efgshg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4045 |
| InitialReportSerialNumber | 11919 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4045, 0)] |
| allReports | [(11919, 4045)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(4045, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 11919) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [11919] |
| reports | [(11919, 4045)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-11-03 12:48:08,337 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-11-03 12:48:17,821 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-11-03 12:48:17,822 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(100) EOB(1) md5(7acedd1a84a4cfcb6e7a16003242945e)

### Sender PCAP

12:48:17.281239 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4045) TExt1 (T=0, L=256)

12:48:18.376782 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment HExt1 (T=0, L=2) RS(4045, 11919) [0 , 100] TExt1 (T=0, L=256)

12:48:18.396717 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment HExt1 (T=0, L=2) RA(11919) TExt1 (T=0, L=256)

### Receiver PCAP

12:48:17.781378 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4045) TExt1 (T=0, L=256)

12:48:17.876723 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment HExt1 (T=0, L=2) RS(4045, 11919) [0 , 100] TExt1 (T=0, L=256)

12:48:18.896797 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment HExt1 (T=0, L=2) RA(11919) TExt1 (T=0, L=256)

## Python to ION Results:

### Sender Commands

#!/bin/bash

# Force the session number to debug authentication.

echo "17" > ltpSessionNumber

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 1, 'xmitKeyID': '"01"', 'xmitKeyMaterial': 'key1.pem'}], 'recvAuthMaterial': [{'recvCiphersuite': 1, 'recvKeyID': '"01"', 'recvKeyMaterial': 'key1.pub'}]}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -S node16.ionsecrc\_1\_tkey1\_rkey2

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4922 |
| InitialReportSerialNumber | 3093 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4922, 0)] |
| allReports | [(3093, 4922)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(4922, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 3093) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [3093] |
| reports | [(3093, 4922)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 18 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1415040115.667241 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 18)

\*\*\* 1415040115.667464 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1415040115.668829 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1415040116.074486 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 18)

\*\*\* 1415040117.170354 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 18)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4922 |
| InitialReportSerialNumber | 3093 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4922, 0)] |
| allReports | [(3093, 4922)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(4922, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 3093) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [3093] |
| reports | [(3093, 4922)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 18 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_1\_tkey1\_rkey2

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_1\_tkey1\_rkey2

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: key3

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '18' ciphersuite '1' key name 'key3' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '1' key name 'key2' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '1' key name 'key1' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

stgs

### Sender PCAP

13:41:56.075355 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4922) TExt1 (T=0, L=256)

13:41:57.097431 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(4922, 3093) [0 , 100] TExt1 (T=0, L=256)

13:41:57.171016 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x9 Report-acknowledgment segment HExt1 (T=0, L=2) RA(3093) TExt1 (T=0, L=256)

### Receiver PCAP

13:41:56.575400 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(4922) TExt1 (T=0, L=256)

13:41:56.597395 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(4922, 3093) [0 , 100] TExt1 (T=0, L=256)

13:41:57.671072 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x9 Report-acknowledgment segment HExt1 (T=0, L=2) RA(3093) TExt1 (T=0, L=256)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  | Ensure: Header / Trailer Extensions | Good: Header / Trailer Extensions | Good: Header / Trailer Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  | Ensure: Authentication uses headers. | Good: Authentication uses headers. | Good: Authentication uses headers. |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(11919, 4045)] | Good: RSN RSN\_CSN [(2332, 7536)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [11919] | Good: Report Acknowledgement(s) sent: [2332] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: No cancel segments sent | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  | Ensure: Trailers | Good: Authentication uses trailers. | Good: Authentication uses trailers. |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  | Ensure: TransmissionRequest.request | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(4045, 0)] | Good: Checkpoint transmitted: [(7536, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  | Ensure: Report transmitted | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  | Ensure: SendReceptionReport | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | Good: 1 | Good: 1 |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  | Ensure: Auth RFC5327 | Good: Auth RFC5327 | Good: Auth RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  | Ensure: Auth May use authentication | Good: Auth May use authentication | Good: Auth May use authentication |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  | Ensure this | Good: verified by examination of configuration files. | Good: verified by examination of configuration files. |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  | Ensure: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  | Ensure this | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  | Can't check this; good auth material | Can't check this; good auth material | Can't check this; good auth material |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  | Ensure: AUTH -- all or none | Good: AUTH -- all or none | Good: AUTH -- all or none |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  | Ensure: AUTH -- extension structure | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (18) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (4045) | Good: InitialCheckpointSerialNumber OK (7536) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 4045 | Good: InitialCheckpointSerialNumber: 7536 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Auth2b

10KB Red data, ciphersuite 1 both directions; bad sender to receiver key.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Like Auth1b but with ciphersuite 1

## Test-Specific Success Criteria

Data segments rejected by receiver.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -S node16.ionsecrc\_1\_tkey1\_rkey2 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 1, 'xmitKeyID': '"01"', 'xmitKeyMaterial': 'key1\_priv.der'}], 'recvAuthMaterial': [{'recvCiphersuite': 1, 'recvKeyID': '"02"', 'recvKeyMaterial': 'key2\_pub.der'}]}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2334 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [2334] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(2334, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 10, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_1\_tkey1\_rkey2

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_1\_tkey1\_rkey2

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: key3

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '18' ciphersuite '1' key name 'key3' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '1' key name 'key2' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '1' key name 'key1' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

efg=g=g=g=g=g=g=g=g=g{gggggggggg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2334 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [2334] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(2334, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 10, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-31 08:53:09,852 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

### Sender PCAP

08:53:18.811543 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:23.810125 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:28.809868 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:33.815408 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:38.816308 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:43.811589 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:48.817596 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:53.818487 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:58.814522 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:54:03.815593 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:54:08.819982 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:13.818643 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:18.825629 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:23.826463 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:28.822199 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:33.827693 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:38.828895 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:43.828340 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:48.834091 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:53.856996 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

### Receiver PCAP

08:53:19.311593 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:24.310197 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:29.309916 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:34.315450 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:39.316365 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:44.311635 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:49.317672 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:54.318536 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:53:59.314569 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:54:04.315639 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(2334) TExt1 (T=0, L=256)

08:54:09.320035 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:14.318701 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:19.325670 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:24.326514 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:29.322237 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:34.327771 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:39.328938 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:44.328377 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:49.334124 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

08:54:54.357060 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

## Python to ION Results:

### Sender Commands

#!/bin/bash

# Force the session number to debug authentication.

echo "17" > ltpSessionNumber

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 1, 'xmitKeyID': '"02"', 'xmitKeyMaterial': 'key2.pem'}], 'recvAuthMaterial': [{'recvCiphersuite': 1, 'recvKeyID': '"02"', 'recvKeyMaterial': 'key2.pub'}]}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -S node16.ionsecrc\_1\_tkey1\_rkey2

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3243 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [3243] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L] |
| cancelReasonCodes | [2] |
| checkpoints | [(3243, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 18 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414760591.806537 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 18)

\*\*\* 1414760591.806689 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414760591.807910 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414760592.214244 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 18)

\*\*\* 1414760634.298739 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 18) reason(2)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 3243 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [3243] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0)] |
| allReports | [] |
| allSegmentsInOrder | [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L] |
| cancelReasonCodes | [2] |
| checkpoints | [(3243, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 18 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_1\_tkey1\_rkey2

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_1\_tkey1\_rkey2

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: key3

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '18' ciphersuite '1' key name 'key3' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '1' key name 'key2' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '1' key name 'key1' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

ssssssssssssssssssssss

### Sender PCAP

09:03:12.213366 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:14.217307 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:16.222516 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:18.227746 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:20.232366 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:22.234682 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:24.240590 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:26.244859 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:28.248927 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:30.253416 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:32.258785 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:34.262646 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:36.265835 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:38.271002 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:40.276256 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:42.279596 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:44.285155 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:46.286803 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:48.289872 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:50.295582 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:52.297100 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:54.335012 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

### Receiver PCAP

09:03:12.713413 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:14.717364 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:16.722585 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:18.727834 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:20.732413 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:22.734774 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:24.740658 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:26.744971 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:28.748975 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:30.753487 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:32.758857 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:34.762720 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:36.765883 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:38.771083 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:40.776294 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:42.779644 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:44.785226 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:46.786837 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:48.789931 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:50.795643 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:52.797142 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(3243) TExt1 (T=0, L=256)

09:03:54.835077 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 12 | Good: Segment Types: 3 , 12 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 12 | Good: Segment Types: 3 , 12 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  | Ensure: Header / Trailer Extensions | Good: Header / Trailer Extensions | Good: Header / Trailer Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  | Ensure: Authentication uses headers. | Good: Authentication uses headers. | Good: Authentication uses headers. |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN Data | Good: CSID Offset Length CSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  | Ensure: No reports transmitted | No Reports [] | No Reports [] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes : 0x02 | Good: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | Good: Authentication uses trailers. | Good: Authentication uses trailers. |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0), (2334, 0)] | Good: Checkpoint transmitted: [(3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0), (3243, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(2334, 10, False)] | Good: checkpoints retransmitted: [(3243, 21, False)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  | Ensure: No reports transmitted | No reports transmitted | No reports transmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: No RedPartReception.indication | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  | Ensure: No GreenPartReception.indication | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No reports sent | No reports sent |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  | Ensure: No report sent. | No report sent. | No report sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | Good: Session cancelled after too many Cancel retransmissions by sender. | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. | No check: both sides have transmission queues. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel ack received (was looking for 13 in response to 12 in [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L]) | No cancel ack received (was looking for 13 in response to 12 in [3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L]) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender stops transmitting data after cancel. | Good: Sender stops transmitting data after cancel. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: No RedPartReception.indication | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: No TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | NO ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | Good: 1 | Good: 1 |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  | Ensure: Auth RFC5327 | Good: Auth RFC5327 | Good: Auth RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  | Ensure: Auth May use authentication | Good: Auth May use authentication | Good: Auth May use authentication |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  | Ensure this | Good: verified by examination of configuration files. | Good: verified by examination of configuration files. |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  | Ensure: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  | Ensure this | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  | Ensure: segments with bad authentication rejected. | Good: segments with bad authentication rejected. | Good: segments with bad authentication rejected. |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  | Ensure: AUTH -- all or none | Good: AUTH -- all or none | Good: AUTH -- all or none |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  | Ensure: AUTH -- extension structure | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (18) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (2334) | Good: InitialCheckpointSerialNumber OK (3243) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 2334 | Good: InitialCheckpointSerialNumber: 3243 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Auth2c

10KB Red data, ciphersuite 1 both directions; bad receiver to sender key.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Like Auth1c but with ciphersuite 1

## Test-Specific Success Criteria

Segments accepted by receiver; report segments rejected by the sender.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -S node16.ionsecrc\_1\_tkey1\_rkey2 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 1, 'xmitKeyID': '"01"', 'xmitKeyMaterial': 'key1\_priv.der'}], 'recvAuthMaterial': [{'recvCiphersuite': 1, 'recvKeyID': '"01"', 'recvKeyMaterial': 'key1\_pub.der'}]}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [3L] |
| InitialCheckpointSerialNumber | 1896 |
| InitialReportSerialNumber | 944 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [944] |
| allCheckpoints | [(1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0)] |
| allReports | [(944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896)] |
| allSegmentsInOrder | [3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 14L, 3L, 14L, 14L, 12L, 14L, 13L, 14L, 14L, 12L, 13L, 14L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(1896, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 14 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | True |
| reports | [(944, 1896)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 22, 9: 0, 10: 0, 11: 0, 12: 2, 13: 2, 14: 7, 15: 0} |
| segmentTypes | [3L, 8L, 12L, 13L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_1\_tkey1\_rkey2

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_1\_tkey1\_rkey2

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: key3

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '18' ciphersuite '1' key name 'key3' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '1' key name 'key2' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '1' key name 'key1' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

efgs

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [3L] |
| InitialCheckpointSerialNumber | 1896 |
| InitialReportSerialNumber | 944 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [944] |
| allCheckpoints | [(1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0), (1896, 0)] |
| allReports | [(944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896), (944, 1896)] |
| allSegmentsInOrder | [3L, 8L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 14L, 3L, 14L, 14L, 14L, 12L, 13L, 14L, 14L, 12L, 13L, 14L, 14L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(1896, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 14 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | True |
| reports | [(944, 1896)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 22, 9: 0, 10: 0, 11: 0, 12: 2, 13: 2, 14: 8, 15: 0} |
| segmentTypes | [3L, 8L, 12L, 13L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-11-03 13:02:33,304 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-11-03 13:02:42,783 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-11-03 13:02:42,784 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 2) len(100) EOB(1) md5(7acedd1a84a4cfcb6e7a16003242945e)

\*\*\* 2014-11-03 13:03:26,893 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-11-03 13:03:26,893 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-11-03 13:03:26,893 - pyltprecv - INFO - (0, 100)

\*\*\* 2014-11-03 13:03:33,011 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-11-03 13:03:33,012 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-11-03 13:03:33,012 - pyltprecv - INFO -

\*\*\* 2014-11-03 13:03:38,016 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-11-03 13:03:38,017 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-11-03 13:03:38,017 - pyltprecv - INFO -

\*\*\* 2014-11-03 13:03:43,024 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-11-03 13:03:43,025 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-11-03 13:03:43,025 - pyltprecv - INFO -

### Sender PCAP

13:02:42.242505 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:02:43.347395 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:45.349155 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:47.351083 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:47.474085 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:02:49.353693 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:51.355434 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:52.475577 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:02:53.358122 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:55.359989 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:57.361835 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:57.476064 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:02:59.363915 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:01.366662 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:02.476783 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:03.368402 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:05.370341 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:07.371788 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:07.477526 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:09.373627 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:11.377204 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:12.478503 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:13.378839 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:15.381922 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:17.383717 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:17.481184 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:19.385327 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:21.387799 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:22.480312 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:23.389478 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:25.393018 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:27.429761 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:27.482346 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:29.430208 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:31.432128 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:32.482316 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:33.434611 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:33.547874 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:35.438129 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:37.440059 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:37.483308 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:38.551924 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:39.441663 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

### Receiver PCAP

13:02:42.742574 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:02:42.847284 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:44.849086 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:46.851025 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:47.974147 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:02:48.853657 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:50.855367 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:52.858066 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:52.975640 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:02:54.859896 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:56.861778 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:02:57.976130 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:02:58.863853 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:00.866626 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:02.868341 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:02.976845 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:04.870277 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:06.871717 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:07.977596 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:08.873558 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:10.877160 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:12.878770 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:12.978566 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:14.881870 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:16.883634 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:17.981260 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:18.885262 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:20.887732 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:22.889435 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:22.980374 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:24.892977 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment HExt1 (T=0, L=2) RS(1896, 944) [0 , 100] TExt1 (T=0, L=256)

13:03:26.929699 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:27.982407 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(1896) TExt1 (T=0, L=256)

13:03:28.930145 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:30.932085 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:32.934544 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:32.982429 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:33.047808 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:34.938036 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:36.940001 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:37.983355 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:38.051877 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:38.941585 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:03:40.943161 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

## Python to ION Results:

### Sender Commands

#!/bin/bash

# Force the session number to debug authentication.

echo "17" > ltpSessionNumber

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 1, 'xmitKeyID': '"01"', 'xmitKeyMaterial': 'key1.pem'}], 'recvAuthMaterial': [{'recvCiphersuite': 1, 'recvKeyID': '"02"', 'recvKeyMaterial': 'key2.pub'}]}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -S node16.ionsecrc\_1\_tkey1\_rkey2

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 10629 |
| InitialReportSerialNumber | 7947 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [7947] |
| allCheckpoints | [(10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0)] |
| allReports | [(7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629)] |
| allSegmentsInOrder | [3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 12L, 13L] |
| cancelReasonCodes | [2] |
| checkpoints | [(10629, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 13 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [(7947, 10629)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 9, 9: 0, 10: 0, 11: 0, 12: 1, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 18 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414429822.990408 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 18)

\*\*\* 1414429822.990545 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414429822.991589 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414429823.398938 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 18)

\*\*\* 1414429865.454935 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 18) reason(2)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99), (0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 10629 |
| InitialReportSerialNumber | 7947 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [7947] |
| allCheckpoints | [(10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0)] |
| allReports | [(7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629)] |
| allSegmentsInOrder | [3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 3L, 3L, 8L, 3L, 3L, 8L, 3L, 12L, 13L] |
| cancelReasonCodes | [2] |
| checkpoints | [(10629, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | 13 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reports | [(7947, 10629)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 9, 9: 0, 10: 0, 11: 0, 12: 1, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 18 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_1\_tkey1\_rkey2

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_1\_tkey1\_rkey2

a ltpxmitauthrule 6 token version

LTP xmit auth rule adding rule keyName: key3

LTP xmit auth rule added for: 18

a ltprecvauthrule 6 token version

a ltprecvauthrule 6 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '18' ciphersuite '1' key name 'key3' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '1' key name 'key2' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '1' key name 'key1' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

stgss+gsss+gss+gss+sgss+gss+gsss+gss+gss}g

### Sender PCAP

13:10:23.397538 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:24.418374 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:25.400278 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:27.402065 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:29.407155 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:29.436454 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:31.408585 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:33.411087 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:34.448704 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:35.415627 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:37.417130 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:39.419858 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:39.438637 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:41.424089 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:43.426635 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:44.440781 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:45.430555 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:47.433429 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:49.435888 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:49.440356 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:51.437082 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:53.442037 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:54.441389 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:55.445440 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:57.446531 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:59.442116 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:59.447999 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:11:01.450728 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:11:03.455075 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:11:04.443159 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:11:05.491359 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:11:06.511662 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=256)

### Receiver PCAP

13:10:23.897602 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:23.918299 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:25.900348 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:27.902108 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:28.936364 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:29.907212 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:31.908634 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:33.911129 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:33.948654 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:35.915704 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:37.917176 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:38.938580 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:39.919924 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:41.924135 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:43.926666 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:43.940697 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:45.930589 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:47.933479 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:48.940300 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:49.935925 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:51.937133 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:53.941317 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:53.942090 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:55.945512 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:57.946599 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:10:58.942022 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:10:59.948065 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:11:01.950792 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:11:03.943116 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x8 Report segment HExt1 (T=0, L=2) RS(10629, 7947) [0 , 100] TExt1 (T=0, L=256)

13:11:03.955142 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=2) Data(0, 100) CP(10629) TExt1 (T=0, L=256)

13:11:05.991425 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0xC Cancel segment from block sender Reason(RLEXC) HExt1 (T=0, L=2) TExt1 (T=0, L=256)

13:11:06.011607 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 18) 0xD Cancel-acknowledgment segment to block sender HExt1 (T=0, L=2) TExt1 (T=0, L=256)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 12 , 13 , 14 | Good: Segment Types: 3 , 8 , 12 , 13 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 8 , 12 , 13 , 14 | Good: Segment Types: 3 , 8 , 12 , 13 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  | Ensure: Header / Trailer Extensions | Good: Header / Trailer Extensions | Good: Header / Trailer Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  | Ensure: Authentication uses headers. | Good: Authentication uses headers. | Good: Authentication uses headers. |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN Data | Good: CSID Offset Length CSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296), (12074, 6296)] | Good: RSN RSN\_CSN [(7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629), (7947, 10629)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes : 0x02 | Good: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  | Ensure: Trailers | Good: Authentication uses trailers. | Good: Authentication uses trailers. |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(6296, 0), (6296, 0), (6296, 0), (6296, 0), (6296, 0), (6296, 0), (6296, 0), (6296, 0), (6296, 0), (6296, 0)] | Good: Checkpoint transmitted: [(10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0), (10629, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  | Ensure: Reports retransmitted | Good: StartReportSegment Timer (Report(s) retransmitted) | Good: StartReportSegment Timer (Report(s) retransmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(6296, 10, 12074)] | Good: checkpoints retransmitted: [(10629, 21, 7947)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  | Ensure: Report transmitted | Good: Report retransmitted | Good: Report retransmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | Good: RLEXC generated. | Good: RLEXC generated. |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  | Ensure: No report sent. | No report ack | No report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | Good: Session Cancelled | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. | No check: both sides have transmission queues. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | Good: CancelACK to sender | Good: CancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No check | No check |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No check: sender can't receive (authenticate) cancel segment. | No check: sender can't receive (authenticate) cancel segment. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | No cancel ack received (was looking for 15 in response to 14 in [3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 8L, 3L, 8L, 8L, 3L, 8L, 8L, 14L, 3L, 14L, 14L, 12L, 14L, 13L, 14L, 14L, 12L, 13L, 14L]) | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: No TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | Good: 1 | Good: 1 |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  | Ensure: Auth RFC5327 | Good: Auth RFC5327 | Good: Auth RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  | Ensure: Auth May use authentication | Good: Auth May use authentication | Good: Auth May use authentication |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  | Ensure this | Good: verified by examination of pcap file. | Good: verified by examination of pcap file. |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  | Ensure: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  | Ensure this | Good: verified by examination of configuration files. | Good: verified by examination of configuration files. |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  | Ensure: segments with bad authentication rejected. | Good: segments with bad authentication rejected. | Good: segments with bad authentication rejected. |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  | Ensure: AUTH -- all or none | Good: AUTH -- all or none | Good: AUTH -- all or none |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  | Ensure: AUTH -- extension structure | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (18) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (6296) | Good: InitialCheckpointSerialNumber OK (10629) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 6296 | Good: InitialCheckpointSerialNumber: 10629 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: Auth3

10KB Red data, ciphersuite 255 in both directions; good keys.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 100 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Like Auth1 but with ciphersuite 255. Note that Ciphersuite 255 is an 'integrity-only' ciphersuite with globally-known authentication material so testing 'bad' authentication material is not possible.

## Test-Specific Success Criteria

Ensure that authentication is invoked and that the block is received correctly.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -S node16.ionsecrc\_255\_255 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 255}], 'recvAuthMaterial': [{'recvCiphersuite': 255}]}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 316 |
| InitialReportSerialNumber | 1850 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(316, 0)] |
| allReports | [(1850, 316)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(316, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 1850) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [1850] |
| reports | [(1850, 316)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_255\_255

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_255\_255

a ltpxmitauthrule 5 token version

LTP xmit auth rule adding rule keyName: (null)

LTP xmit auth rule added for: 16

a ltpxmitauthrule 5 token version

LTP xmit auth rule adding rule keyName: (null)

LTP xmit auth rule added for: 18

a ltprecvauthrule 5 token version

a ltprecvauthrule 5 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '16' ciphersuite '255' key name '' frequency '0'

Transmission rule: ltp engine id '18' ciphersuite '255' key name '' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '255' key name '' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '255' key name '' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

efgshg

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 316 |
| InitialReportSerialNumber | 1850 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(316, 0)] |
| allReports | [(1850, 316)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(316, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 1850) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [1850] |
| reports | [(1850, 316)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-27 13:25:42,654 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-27 13:25:52,127 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-27 13:25:52,128 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 1) len(100) EOB(1) md5(7acedd1a84a4cfcb6e7a16003242945e)

### Sender PCAP

13:25:51.582571 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=1) Data(0, 100) CP(316) TExt1 (T=0, L=10)

13:25:52.638915 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment HExt1 (T=0, L=1) RS(316, 1850) [0 , 100] TExt1 (T=0, L=10)

13:25:52.641391 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment HExt1 (T=0, L=1) RA(1850) TExt1 (T=0, L=10)

### Receiver PCAP

13:25:52.082638 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=1) Data(0, 100) CP(316) TExt1 (T=0, L=10)

13:25:52.138775 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment HExt1 (T=0, L=1) RS(316, 1850) [0 , 100] TExt1 (T=0, L=10)

13:25:53.141443 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment HExt1 (T=0, L=1) RA(1850) TExt1 (T=0, L=10)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'xmitAuthMaterial': [{'xmitCiphersuite': 255}], 'recvAuthMaterial': [{'recvCiphersuite': 255}]}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver -S node16.ionsecrc\_255\_255

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 8337 |
| InitialReportSerialNumber | 2113 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(8337, 0)] |
| allReports | [(2113, 8337)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(8337, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 2113) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [2113] |
| reports | [(2113, 8337)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 57 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414430942.229434 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 57)

\*\*\* 1414430942.229583 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414430942.231319 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414430942.435345 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 57)

\*\*\* 1414430943.449346 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 57)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 99)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 99 |
| EORP | 99 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 8337 |
| InitialReportSerialNumber | 2113 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(8337, 0)] |
| allReports | [(2113, 8337)] |
| allSegmentsInOrder | [3L, 8L, 9L] |
| cancelReasonCodes | [] |
| checkpoints | [(8337, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [0] |
| highestByteIndex | 99 |
| highestRedByteIndex | 99 |
| lastCancelOrAck | None |
| lastReportOrAck | ('Ack', 2113) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 1 |
| maxTrailerExtensions | 1 |
| receiverCancelled | False |
| reportAcks | [2113] |
| reports | [(2113, 8337)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 0, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 1, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [3L, 8L, 9L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 57 |
| sessionOriginator | 18 |
| trailerExtensions | [0, 0, 0] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: S

-S was specified; using security file node16.ionsecrc\_255\_255

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

Using ionsecrc file node16.ionsecrc\_255\_255

a ltpxmitauthrule 5 token version

LTP xmit auth rule adding rule keyName: (null)

LTP xmit auth rule added for: 16

a ltpxmitauthrule 5 token version

LTP xmit auth rule adding rule keyName: (null)

LTP xmit auth rule added for: 18

a ltprecvauthrule 5 token version

a ltprecvauthrule 5 token version

Listing ltpxmitauthrules...

Transmission rule: ltp engine id '16' ciphersuite '255' key name '' frequency '0'

Transmission rule: ltp engine id '18' ciphersuite '255' key name '' frequency '0'

Listing ltprecvauthrules...

Receiving rule: ltp engine id '16' ciphersuite '255' key name '' frequency '0'

Receiving rule: ltp engine id '18' ciphersuite '255' key name '' frequency '0'

Stopping ionsecadmin.

Stopping ltpadmin.

stgs

### Sender PCAP

13:29:02.434545 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 57) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=1) Data(0, 100) CP(8337) TExt1 (T=0, L=10)

13:29:03.437306 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 57) 0x8 Report segment HExt1 (T=0, L=1) RS(8337, 2113) [0 , 100] TExt1 (T=0, L=10)

13:29:03.448986 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 57) 0x9 Report-acknowledgment segment HExt1 (T=0, L=1) RA(2113) TExt1 (T=0, L=10)

### Receiver PCAP

13:29:02.934585 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 57) 0x3 Red data, Checkpoint, EORP, EOB C(6) HExt1 (T=0, L=1) Data(0, 100) CP(8337) TExt1 (T=0, L=10)

13:29:02.937236 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 57) 0x8 Report segment HExt1 (T=0, L=1) RS(8337, 2113) [0 , 100] TExt1 (T=0, L=10)

13:29:03.949067 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 57) 0x9 Report-acknowledgment segment HExt1 (T=0, L=1) RA(2113) TExt1 (T=0, L=10)

## Comments:

None

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 3 , 8 , 9 | Good: Segment Types: 3 , 8 , 9 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  | Ensure: Header / Trailer Extensions | Good: Header / Trailer Extensions | Good: Header / Trailer Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  | Ensure: Authentication uses headers. | Good: Authentication uses headers. | Good: Authentication uses headers. |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(1850, 316)] | Good: RSN RSN\_CSN [(2113, 8337)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [1850] | Good: Report Acknowledgement(s) sent: [2113] |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) |  | No Cancel segments sent | No Cancel segments sent |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  | Ensure: Trailers | Good: Authentication uses trailers. | Good: Authentication uses trailers. |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(316, 0)] | Good: Checkpoint transmitted: [(8337, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  | Ensure: Reports transmitted | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  | Ensure: Report transmitted | No report retransmitted (though report transmitted) | No report retransmitted (though report transmitted) |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No Report segments retransmitted | No Report segments retransmitted |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No RLEXC due to report retransmission | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report after report ack | Good: no report after report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No cancel segments sent (startCancelTimer) | No cancel segments sent (startCancelTimer) |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | No cancel segment retransmission | No cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No cancellation segments | No cancellation segments |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel segments sent (stopCancelTimer) | No cancel segments sent (stopCancelTimer) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No cancel segments sent (cancelSession) | No cancel segments sent (cancelSession) |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  | Ensure: RedPartReception.indication | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  | Ensure: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | NO ReceptionSessionCancellation.indication | NO ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | Good: 1 | Good: 1 |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  | Ensure: Auth RFC5327 | Good: Auth RFC5327 | Good: Auth RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  | Ensure: Auth May use authentication | Good: Auth May use authentication | Good: Auth May use authentication |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  | Ensure this | Good: verified by examination of pcap file. | Good: verified by examination of pcap file. |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  | Ensure: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified | Good: AUTH -- no cookies; verified |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  | Ensure this | Good: verified by examination of configuration files. | Good: verified by examination of configuration files. |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  | Can't check this; good auth material | Can't check this; good auth material | Can't check this; good auth material |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  | Ensure: AUTH -- all or none | Good: AUTH -- all or none | Good: AUTH -- all or none |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  | Ensure: AUTH -- extension structure | Good: AUTH -- extension structure | Good: AUTH -- extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) | Good: SessionNumber OK (57) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (316) | Good: InitialCheckpointSerialNumber OK (8337) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 316 | Good: InitialCheckpointSerialNumber: 8337 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: RLEXC\_S

Block reception of Red data at receiver and let transmitter keep transmitting checkpoints until it gives up.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 10000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Block reception at receiver; wait for sender to give up and cancel the session after (re)transmitting a bunch of checkpoints; receiver never gets anything.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -i node16.ionrc -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py \

-E 16 \

-n "{18: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -s 10.0.0.2 -t 500

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146), (7147, 8337), (8338, 9528), (9529, 9999), (9529, 9999), (9529, 9999), (9529, 9999), (9529, 9999), (9529, 9999), (9529, 9999), (9529, 9999), (9529, 9999), (9529, 9999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4801 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [4801] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4801, 0), (4801, 0), (4801, 0), (4801, 0), (4801, 0), (4801, 0), (4801, 0), (4801, 0), (4801, 0), (4801, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L, 12L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(4801, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 10, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 10, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

Processing option: i

Using ionrc file node16.ionrc

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeefggggggggg=g=g=g=g=g=g=g=g=g{gggggggggg

### Receiver Summary:

No automated checks for this test.

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-30 10:39:33,421 - pyltprecv - INFO - pyltprecv LTP engine 16 listening on ('', 1113) client Service 6

### Sender PCAP

10:39:41.348423 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

10:39:42.339153 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

10:39:43.321842 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

10:39:44.304878 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

10:39:45.287849 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

10:39:46.270833 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

10:39:47.253771 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7147, 1191)

10:39:48.236728 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8338, 1191)

10:39:49.219784 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:39:54.346699 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:39:59.347589 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:40:04.348403 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:40:09.349275 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:40:14.350130 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:40:19.350949 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:40:24.351747 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:40:29.352741 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:40:34.353463 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(4801)

10:40:39.355242 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:40:44.355839 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:40:49.356745 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:40:54.357714 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:40:59.358649 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:41:04.359642 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:41:09.360558 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:41:14.361721 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:41:19.362699 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

10:41:24.363570 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xC Cancel segment from block sender Reason(RLEXC)

### Receiver PCAP

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -s 10.0.0.2 -t 500

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 5925), (5926, 7110), (7111, 8295), (8296, 9480), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999), (9481, 9999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 4701 |
| InitialReportSerialNumber | False |
| MiscoloredData | False |
| ReceiveIndications | [] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [4701] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0), (4701, 0)] |
| allReports | [] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 12L] |
| cancelReasonCodes | [2] |
| checkpoints | [(4701, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | 12 |
| lastReportOrAck | None |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reports | [] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 21, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0, 9: 0, 10: 0, 11: 0, 12: 1, 13: 0, 14: 0, 15: 0} |
| segmentTypes | [0L, 3L, 12L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 32 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414174611.980575 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 32)

\*\*\* 1414174611.981077 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414174611.982686 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414174622.150641 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 32)

\*\*\* 1414174664.262814 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 32) reason(2)

### Receiver Summary:

No automated checks for this test.

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

### Sender PCAP

14:16:53.193294 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

14:16:54.401596 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

14:16:55.609538 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

14:16:56.816714 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

14:16:58.023000 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4741, 1185)

14:16:59.231092 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5926, 1185)

14:17:00.440088 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7111, 1185)

14:17:01.647034 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8296, 1185)

14:17:02.151304 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:04.162955 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:06.167690 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:08.174170 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:10.180691 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:12.184730 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:14.188669 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:16.193460 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:18.200915 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:20.205732 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:22.209904 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:24.214582 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:26.221560 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:28.227739 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:30.234419 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:32.235941 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:34.240328 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:36.244667 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:38.250925 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:40.255224 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:42.258099 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(4701)

14:17:44.268009 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 32) 0xC Cancel segment from block sender Reason(RLEXC)

### Receiver PCAP

## Comments:

### ION to Python Comments

The Python implementation transmits the checkpoints a number of times, then the cancel segment a number of times, THEN scrubs.

### Python to ION Comments

The ION implementation transmits the checkpoint a numer of times, then cancels with RLEXC and quits.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: SegmentTypes: 0, 3, 12 | Good: SegmentTypes: 0, 3, 12 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: SegmentTypes: 0, 3, 12 | Good: SegmentTypes: 0, 3, 12 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: Offset, Length, CSID, Data | Good: Offset, Length, CSID, Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | No report segments. | No report segments. |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted | Good: Checkpoint transmitted |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check; receiver can transmit (as far as the sender is concerned). | No check; receiver can transmit (as far as the sender is concerned). |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | No report segments | No report segments |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No report segments | No report segments |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: Checkpoint retransmissions seen. | Good: Checkpoint retransmissions seen. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | Good: Sender eventually gives up and cancels. | Good: Sender eventually gives up and cancels. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | No report segments | No report segments |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No report segments | No report segments |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No data received | No data received |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No data received | No data received |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | No data received | No data received |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | No data received | No data received |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No Report segments sent. | No Report segments sent. |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No Report segments sent. | No Report segments sent. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission | No cancel segment retransmission. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | No check; peer can transmit. | No check; peer can transmit. |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | Good: Sender eventually gives up with RLEXC | Good: Sender eventually gives up with RLEXC |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Retransmissions of cancel segments seen. | No retransmission of cancel segments seen. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No Cancel Acknowledgement sent. | No Cancel Acknowledgement sent. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | No Cancel Acknowledgement sent. | No Cancel Acknowledgement sent. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No Cancel Acknowledgement sent. | No Cancel Acknowledgement sent. |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No Cancel Acknowledgement sent. | No Cancel Acknowledgement sent. |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  | Ensure that sender gives up and cancels session. | Good: Sender gives up. | Good: Sender gives up. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | No RedPartReception.indication | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | No segments received | No segments received |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No segments received | No segments received |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK | Good: SessionNumber OK |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK | Good: InitialCheckpointSerialNumber OK |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 4801 | Good: InitialCheckpointSerialNumber: 4701 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: RLEXC\_R

Block reception of Red data after transmission of report but before report-acknowledgement; wait for receiver to give up after retransmitting a bunch of reports.

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 0 | (plr) |
| RED\_SIZE | 10000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 1000 | (Bytes/sec) |
| SEG\_SIZE | 1200 | (Bytes) |

## Test Procedure

Drop the link in at least the forward direction after the first report but before the report acknowledgement segment ; provide no link queue to either the sender or receiver (elicit retransmissions of reports (from receiver) segments; cause retransmission limit exceeded RLEXC)

## Test-Specific Success Criteria

Report segments are retransmitted by the sender until the retransmission limit is exceeded. When the retransmission limit is exceeded the session is terminated and cancellation is sent.

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 --maxRxmtCycRecv 2 -n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -T 8 -R 14 --skip 1 -s 10.0.0.2 -t 500

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146), (7147, 8337), (8338, 9528), (9529, 9999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2753 |
| InitialReportSerialNumber | 15602 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(2753, 0)] |
| allReports | [(15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(2753, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | 14 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [15602] |
| reports | [(15602, 2753)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 22, 9: 1, 10: 0, 11: 0, 12: 0, 13: 0, 14: 11, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 9L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeefgggggggggshgssssssssssssssssssssssssssssssss

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1191), (1192, 2382), (2383, 3573), (3574, 4764), (4765, 5955), (5956, 7146), (7147, 8337), (8338, 9528), (9529, 9999)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 2753 |
| InitialReportSerialNumber | 15602 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [15602] |
| allCheckpoints | [(2753, 0)] |
| allReports | [(15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(2753, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | 14 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [(15602, 2753)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 22, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 11, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 2 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-24 14:36:21,870 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-24 14:36:30,344 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 2)

\*\*\* 2014-10-24 14:36:38,194 - pyltprecv - INFO - RedPartReception.indication Red Part Reception (16, 2) len(10000) EOB(1) md5(dc50add066871756c3f0260f0aa76cd2)

\*\*\* 2014-10-24 14:37:22,263 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 2) reason RLEXC

\*\*\* 2014-10-24 14:37:22,263 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-24 14:37:22,264 - pyltprecv - INFO - (0, 10000)

### Sender PCAP

14:36:29.805839 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:36:30.788614 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

14:36:31.771542 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

14:36:32.754163 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

14:36:33.737312 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

14:36:34.720386 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

14:36:35.703205 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7147, 1191)

14:36:36.686094 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8338, 1191)

14:36:37.669301 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(2753)

14:36:38.707372 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:38.710966 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x9 Report-acknowledgment segment RA(15602)

14:36:40.711930 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:42.715790 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:44.718362 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:46.720822 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:48.720930 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:50.724750 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:52.725375 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:54.728262 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:56.732232 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:58.735011 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:00.738322 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:02.740127 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:04.742587 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:06.745275 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:08.748028 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:10.750593 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:12.753252 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:14.755486 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:16.758117 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:18.759679 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:20.762965 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:22.767895 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:24.769884 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:26.774151 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:28.776190 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:30.779918 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:32.783608 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:34.785881 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:36.789016 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:38.793968 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:40.795855 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:42.797367 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

### Receiver PCAP

14:36:30.305907 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:36:31.288661 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1191)

14:36:32.271619 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2383, 1191)

14:36:33.254220 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3574, 1191)

14:36:34.237385 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4765, 1191)

14:36:35.220446 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5956, 1191)

14:36:36.203253 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7147, 1191)

14:36:37.186157 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8338, 1191)

14:36:38.169383 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9529, 471) CP(2753)

14:36:38.207309 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:40.211854 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:42.215711 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:44.218295 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:46.220733 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:48.220885 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:50.224684 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:52.225312 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:54.228204 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:56.232172 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:36:58.234945 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:00.238250 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:02.240056 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:04.242503 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:06.245201 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:08.247957 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:10.250535 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:12.253182 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:14.255407 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:16.257988 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:18.259618 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:20.262903 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0x8 Report segment RS(2753, 15602) [0 , 10000]

14:37:22.267822 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:24.269823 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:26.274086 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:28.276122 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:30.279850 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:32.283550 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:34.285743 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:36.288955 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:38.293909 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:40.295790 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

14:37:42.297301 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 2) 0xE Cancel segment from block receiver Reason(RLEXC)

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 -R 1000 -s 1200 \

-r file:redPart \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

Python\_Files/supportFiles/ebtables\_block\_interfaces.py -i br0 -T 8 -R 14 --skip 1 -s 10.0.0.2 -t 500

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 5925), (5926, 7110), (7111, 8295), (8296, 9480), (9481, 9999)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1484 |
| InitialReportSerialNumber | 5251 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1484, 0)] |
| allReports | [(5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L, 8L, 9L, 8L, 9L, 8L, 9L, 8L, 9L, 8L, 9L, 8L, 8L, 8L, 8L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(1484, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | 14 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [5251] |
| reports | [(5251, 1484)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 10, 9: 6, 10: 0, 11: 0, 12: 0, 13: 0, 14: 10, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 9L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 9 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414178129.337061 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 9)

\*\*\* 1414178129.337197 - pyltpsend - INFO - Setting send rate control to 1000.000000 bytes per second.

\*\*\* 1414178129.338326 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414178139.513041 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 9)

\*\*\* 1414178140.566236 - pyltpsend - INFO - TransmissionSessionCompletion.indication: Transmitter reports transmission session completion for session: (18, 9)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1185), (1186, 2370), (2371, 3555), (3556, 4740), (4741, 5925), (5926, 7110), (7111, 8295), (8296, 9480), (9481, 9999)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 9999 |
| EORP | 9999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1484 |
| InitialReportSerialNumber | 5251 |
| MiscoloredData | False |
| ReceiveIndications | ['RedPartReception', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCompletion'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [5251] |
| allCheckpoints | [(1484, 0)] |
| allReports | [(5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L] |
| cancelReasonCodes | [2, 2, 2, 2, 2, 2, 2, 2, 2, 2] |
| checkpoints | [(1484, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 9999 |
| highestRedByteIndex | 9999 |
| lastCancelOrAck | 14 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reports | [(5251, 1484)] |
| sdnvSizes | [1, 2] |
| segmentTypeCounts | {0: 8, 1: 0, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 10, 9: 0, 10: 0, 11: 0, 12: 0, 13: 0, 14: 10, 15: 0} |
| segmentTypes | [0L, 3L, 8L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 9 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

ssssssssstg+g+g+g+g+g+g+g+g+g[gggggggggg

### Sender PCAP

15:15:30.548465 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

15:15:31.757868 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

15:15:32.965883 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

15:15:34.173439 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

15:15:35.381635 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4741, 1185)

15:15:36.590131 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5926, 1185)

15:15:37.798243 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7111, 1185)

15:15:39.007092 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8296, 1185)

15:15:39.512537 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(1484)

15:15:40.517278 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:15:40.563579 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(5251)

15:15:45.749702 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:15:45.790452 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(5251)

15:15:50.752556 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:15:50.761574 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(5251)

15:15:55.753588 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:15:55.782839 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(5251)

15:16:00.753945 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:00.803906 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(5251)

15:16:05.755342 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:05.774312 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x9 Report-acknowledgment segment RA(5251)

15:16:10.756107 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:15.757086 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:20.758390 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:25.759314 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:30.761718 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:35.762751 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:40.764051 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:45.765268 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:50.766450 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:55.767530 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:17:00.768973 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:17:05.770124 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:17:10.771122 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:17:15.772337 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

### Receiver PCAP

15:15:31.048536 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1186)

15:15:32.257934 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1186, 1185)

15:15:33.465953 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2371, 1185)

15:15:34.673505 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3556, 1185)

15:15:35.881706 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4741, 1185)

15:15:37.090198 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5926, 1185)

15:15:38.298304 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7111, 1185)

15:15:39.507194 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8296, 1185)

15:15:40.012598 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(9481, 519) CP(1484)

15:15:40.017230 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:15:45.249629 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:15:50.252480 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:15:55.253520 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:00.253871 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:05.255270 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:10.256067 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:15.257009 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:20.258315 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:25.259264 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0x8 Report segment RS(1484, 5251) [0 , 10000]

15:16:30.261657 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:35.262685 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:40.263986 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:45.265197 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:50.266383 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:16:55.267451 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:17:00.268886 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:17:05.270049 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:17:10.271063 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

15:17:15.272253 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 9) 0xE Cancel segment from block receiver Reason(RLEXC)

## Comments:

### ION to Python Comments

Because of the way the link is blocked, the sender sends a report acknowledgement, then gets a bunch of (to it, spurious) reports after that, then the cancel segment.  
  
Because the session no longer exists (report ack has been sent), the spurious reports AND the cancellation segment are ignored.

### Python to ION Comments

Because of the way the link is blocked, the sender sends a report acknowledgement, then gets a bunch of (to it, spurious) reports after that, then the cancel segment.  
  
Because the session no longer exists (report ack has been sent), the spurious reports AND the cancellation segment are ignored.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 3 , 8 , 14 | Good: Segment Types: 0 , 3 , 8 , 14 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 3 , 8 , 14 | Good: Segment Types: 0 , 3 , 8 , 14 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753), (15602, 2753)] | Good: RSN RSN\_CSN [(5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484), (5251, 1484)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | No report Acknowledgements | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 | Good: CancelReasonCodes: 0x02 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  | No cancellation request primitive attempted | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(2753, 0)] | Good: Checkpoint transmitted: [(1484, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | Good: StartReportSegment Timer (Report(s) retransmitted) | Good: StartReportSegment Timer (Report(s) retransmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No report segments | No report segments |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | No checkpoints retransmitted through checkpoints transmitted. | No checkpoints retransmitted through checkpoints transmitted. |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected. | No RXMTCYCEXC detected. |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | Good: Report retransmitted | Good: Report retransmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  | Ensure: retransmission or reports results in cancel(RLEXC) | Good: retransmission of reports elicits cancel with reason RLEXC | Good: retransmission of reports elicits cancel with reason RLEXC |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  | Ensure: Reports retransmitted until RLEXC sent by receiver. | Good: Reports retransmitted until RLEXC sent. | Good: Reports retransmitted until RLEXC sent. |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | No report ack | No report ack |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | Good: Receiver eventually gives up with RLEXC | Good: Receiver eventually gives up with RLEXC |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | See Note | See Note |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender | NO cancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No cancel ack received (was looking for 15 in response to 14 in [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 8L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L]) | No cancel ack received (was looking for 15 in response to 14 in [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L, 8L, 9L, 8L, 9L, 8L, 9L, 8L, 9L, 8L, 9L, 8L, 8L, 8L, 8L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L]) |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | No check: session ends normally at sender. | No check: session ends normally at sender. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  | Verify that additional reports and cancel segments are ignored after report-ack is sent. | Good: additional reports and cancel segments for non-extant session are ignored. | Good: additional reports and cancel segments for non-extant session are ignored. |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | Good: RedPartReception.indication | Good: RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | Good: TransmissionSessionCompletion.indication | Good: TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | NO TransmissionSessionCancellation.indication | NO TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (2) | Good: SessionNumber OK (9) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (2753) | Good: InitialCheckpointSerialNumber OK (1484) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 2753 | Good: InitialCheckpointSerialNumber: 1484 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP | Good: CCSDS ENCAP |

# Test: RXMTCYCEXC\_S

Retransmission cyclesexceeded at sender (rexmitting checkpoints).

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 50 | (plr) |
| RED\_SIZE | 20000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 10000 | (Bytes/sec) |
| SEG\_SIZE | 1000 | (Bytes) |

## Test Procedure

Turn the error rate up, RedPart size up, and segment size down and then jack with the MAXRXMTCYC value for the transmitter (set it significantly below that of the receiver)

## There Are No ION to Python Results:

There are no results for this test.

## Python to ION Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltpsend.py -E 18 -e 16 -A 127.0.0.1 -P 1113 --rate 10000 --segmentSize 1000 \

-r file:redPart \

--maxRxmtCycSend 3 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True, 'OldIONLTPOffsetFix': False}}" &

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

echo "iptables rules for foo container" > foo\_iptables.txt

/usr/sbin/iptables -nn -v --list >> foo\_iptables.txt

echo "end of iptables rules for foo container" > foo\_iptables.txt

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPReceiver

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 985), (986, 1970), (1971, 2955), (2956, 3940), (3941, 4925), (4926, 5910), (5911, 6895), (6896, 7880), (7881, 8865), (8866, 9850), (9851, 10835), (10836, 11820), (11821, 12805), (12806, 13790), (13791, 14775), (14776, 15760), (15761, 16745), (16746, 17729), (17730, 18713), (18714, 19697), (19698, 19999), (0, 985), (2956, 3940), (4926, 5910), (5911, 6895), (8866, 9850), (9851, 10835), (11821, 12805), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (0, 985), (2956, 3940), (4926, 5910), (5911, 6895), (4926, 5910)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 19999 |
| EORP | 19999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 9096 |
| InitialReportSerialNumber | 7971 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [9099] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(9096, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9098, 0), (9099, 0)] |
| allReports | [(7971, 9096), (7972, 9097), (7973, 9098)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 8L, 9L, 0L, 0L, 0L, 1L, 8L, 9L, 1L, 12L, 12L, 13L] |
| cancelReasonCodes | [5, 5] |
| checkpoints | [(9096, 0), (9097, 0), (9098, 0), (9099, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 5910 |
| highestRedByteIndex | 19999 |
| lastCancelOrAck | 13 |
| lastReportOrAck | ('Ack', 7973) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [7971, 7972, 7973] |
| reports | [(7971, 9096), (7972, 9097), (7973, 9098)] |
| sdnvSizes | [1, 2, 3] |
| segmentTypeCounts | {0: 30, 1: 13, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 3, 9: 3, 10: 0, 11: 0, 12: 2, 13: 1, 14: 0, 15: 0} |
| segmentTypes | [0L, 1L, 3L, 8L, 9L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 65 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### log\_pyltpsend

\*\*\* 1414160505.004362 - pyltpsend - INFO - TransmissionSessionStart.indication: New transmit session started with session ID: (18, 65)

\*\*\* 1414160505.004587 - pyltpsend - INFO - Setting send rate control to 10000.000000 bytes per second.

\*\*\* 1414160505.006177 - pyltpsend - INFO - Sender userCancelAfter is None

\*\*\* 1414160507.132570 - pyltpsend - INFO - InitialTransmissionCompletion.indication: Transmitter reports initial transmission completion for session: (18, 65)

\*\*\* 1414160530.481220 - pyltpsend - INFO - TransmissionSessionCancellation.indication Transmitting application reports cancellation of session ID: (18, 65) reason(5)

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(986, 1970), (1971, 2955), (3941, 4925), (6896, 7880), (7881, 8865), (10836, 11820), (12806, 13790), (13791, 14775), (14776, 15760), (15761, 16745), (16746, 17729), (18714, 19697), (19698, 19999), (8866, 9850), (9851, 10835), (11821, 12805), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (17730, 18713), (0, 985), (2956, 3940), (5911, 6895)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 19999 |
| EORP | 19999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 9096 |
| InitialReportSerialNumber | 7971 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [7973] |
| allCheckpoints | [(9096, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9098, 0)] |
| allReports | [(7971, 9096), (7972, 9097), (7972, 9097), (7972, 9097), (7972, 9097), (7972, 9097), (7973, 9098)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L, 0L, 0L, 0L, 1L, 8L, 1L, 1L, 8L, 1L, 8L, 8L, 1L, 1L, 8L, 9L, 0L, 0L, 1L, 8L, 12L, 13L, 12L, 13L] |
| cancelReasonCodes | [5, 5] |
| checkpoints | [(9096, 0), (9097, 0), (9098, 0)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 6895 |
| highestRedByteIndex | 19999 |
| lastCancelOrAck | 13 |
| lastReportOrAck | Report |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | False |
| reportAcks | [7971, 7972] |
| reports | [(7971, 9096), (7972, 9097), (7973, 9098)] |
| sdnvSizes | [1, 2, 3] |
| segmentTypeCounts | {0: 17, 1: 7, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 7, 9: 2, 10: 0, 11: 0, 12: 2, 13: 2, 14: 0, 15: 0} |
| segmentTypes | [0L, 1L, 3L, 8L, 9L, 12L, 13L] |
| segmentVersions | [0L] |
| senderCancelled | True |
| sessionNumber | 65 |
| sessionOriginator | 18 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

sssssssssssssgsssssgss+gs+g+gss+gssssgs}gsg

### Sender PCAP

10:21:45.109141 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 986)

10:21:45.209211 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(986, 985)

10:21:45.309958 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1971, 985)

10:21:45.409600 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2956, 985)

10:21:45.512825 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3941, 985)

10:21:45.612730 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4926, 985)

10:21:45.713440 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5911, 985)

10:21:45.814526 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(6896, 985)

10:21:45.914697 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7881, 985)

10:21:46.016206 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8866, 985)

10:21:46.119151 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(9851, 985)

10:21:46.220494 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(10836, 985)

10:21:46.321104 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(11821, 985)

10:21:46.421268 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(12806, 985)

10:21:46.522968 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(13791, 985)

10:21:46.623500 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(14776, 985)

10:21:46.724225 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(15761, 985)

10:21:46.825366 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(16746, 984)

10:21:46.925462 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(17730, 984)

10:21:47.026281 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(18714, 984)

10:21:47.130088 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(19698, 302) CP(9096)

10:21:48.132759 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9096, 7971) [0 , 20000] / Raw

10:21:48.174837 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x9 Report-acknowledgment segment RA(7971)

10:21:48.184441 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 986)

10:21:48.187318 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2956, 985)

10:21:48.190310 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4926, 985)

10:21:48.191863 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5911, 985)

10:21:48.194993 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8866, 985)

10:21:48.196250 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(9851, 985)

10:21:48.199198 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(11821, 985)

10:21:48.203320 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:50.209280 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:52.213879 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:54.220043 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:56.224229 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:58.227283 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:22:00.233064 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:22:02.239417 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:22:04.244853 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:22:06.248475 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:22:08.254513 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:22:09.417376 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9097, 7972) [0 , 18714]

10:22:09.433682 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x9 Report-acknowledgment segment RA(7972)

10:22:09.442924 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 986)

10:22:09.445595 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2956, 985)

10:22:09.449584 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4926, 985)

10:22:09.450942 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(5911, 985) CP(9098)

10:22:10.453468 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9098, 7973) [0 , 6896]

10:22:10.472789 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x9 Report-acknowledgment segment RA(7973)

10:22:10.482511 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(4926, 985) CP(9099)

10:22:10.484574 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0xC Cancel segment from block sender Reason(RXMTCYCEXC)

10:22:12.488126 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0xC Cancel segment from block sender Reason(RXMTCYCEXC)

10:22:13.490364 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0xD Cancel-acknowledgment segment to block sender

### Receiver PCAP

10:21:45.709278 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(986, 985)

10:21:45.810029 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1971, 985)

10:21:46.012973 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3941, 985)

10:21:46.314595 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(6896, 985)

10:21:46.414756 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7881, 985)

10:21:46.720587 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(10836, 985)

10:21:46.921318 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(12806, 985)

10:21:47.023032 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(13791, 985)

10:21:47.123564 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(14776, 985)

10:21:47.224283 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(15761, 985)

10:21:47.325423 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(16746, 984)

10:21:47.526340 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(18714, 984)

10:21:47.630144 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(19698, 302) CP(9096)

10:21:47.632704 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9096, 7971) [0 , 20000] / Raw

10:21:48.674957 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x9 Report-acknowledgment segment RA(7971)

10:21:48.695034 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8866, 985)

10:21:48.696285 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(9851, 985)

10:21:48.699237 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(11821, 985)

10:21:48.703358 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:48.705976 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9097, 7972) [0 , 18714]

10:21:50.709342 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:52.713939 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:53.914273 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9097, 7972) [0 , 18714]

10:21:56.724293 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:21:58.915270 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9097, 7972) [0 , 18714]

10:22:03.916346 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9097, 7972) [0 , 18714]

10:22:06.748569 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:22:08.754592 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(17730, 984) CP(9097)

10:22:08.917299 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9097, 7972) [0 , 18714]

10:22:09.933743 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x9 Report-acknowledgment segment RA(7972)

10:22:09.942976 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 986)

10:22:09.945635 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2956, 985)

10:22:09.950979 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(5911, 985) CP(9098)

10:22:09.953418 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0x8 Report segment RS(9098, 7973) [0 , 6896]

10:22:10.984627 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0xC Cancel segment from block sender Reason(RXMTCYCEXC)

10:22:10.989665 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0xD Cancel-acknowledgment segment to block sender

10:22:12.988183 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0xC Cancel segment from block sender Reason(RXMTCYCEXC)

10:22:12.990314 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(18, 65) 0xD Cancel-acknowledgment segment to block sender

## Comments:

### ION to Python Comments

Hard to elicit this behavior from the ION implementation. The Python implementation allows the retransmission cycle limit to be set from the command line.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  |  | Good: Segment Types: 0 , 1 , 3 , 8 , 9 , 12 , 13 |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  |  | Good: Segment Types: 0 , 1 , 3 , 8 , 9 , 12 , 13 |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  |  | No Extensions |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  |  | No Extensions |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  |  | Good: CSID Offset Length CSN CSN\_RSN Data |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  |  | Good: RSN [(7971, 9096), (7972, 9097), (7972, 9097), (7972, 9097), (7972, 9097), (7972, 9097), (7973, 9098)] |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  |  | No report Acknowledgements |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x05 |  | Good: CancelReasonCodes: 0x05 |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  |  | No Trailers |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  |  | Good: TransmissionRequest.request |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  |  |  | No cancellation request primitive attempted |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  |  | No corrupt segments. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  |  | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  |  | No check -- receive client service exists. |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  |  | Good: StartTransmission |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  |  | Good: Checkpoint transmitted: [(9096, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9097, 0), (9098, 0), (9099, 0)] |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  |  | No check: receiver is transmitting. |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  |  | Good: StartReportSegment Timer (Report(s) retransmitted) |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  |  | No check -- receiver is transmitting |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  |  | No Link Cues |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  | Look for lots of retransmitted checkpoints. |  | Good: checkpoints retransmitted: [(9096, 1, 7971), (9097, 11, 7972), (9098, 1, 7973), (9099, 1, False)] |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  | Ensure: retransmission limit for checkpoint expired, sender issues cancel with RXMTCYCEXC |  | Good: retransmission limit for checkpoint expired, sender issues cancel with RXMTCYCEXC |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  |  | Good: Report retransmitted |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  |  | No cancel due to RLEXC |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  |  | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  |  | No GreenPartReception.indication |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  |  | Good: SendReceptionReport |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  |  | No RLEXC due to report retransmission |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  |  | Good: InitialTransmissionCompletion.indication |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  |  | No red data retransmission |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  |  | Good: no report with same RSN sent after report ack for that RSN. |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  |  | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  |  | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  |  | No session cancellation due to too many cancel segment retransmissions |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  |  | Good: Cancel segment retransmission |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  |  | No check: both sides have transmission queues. |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  |  | Good: CancelACK to sender |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  |  | No cancelACK to receiver |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  |  | Good: no cancel sent after cancel ack received. |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  |  | Good: Sender gives up. |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  |  | CHECKME: closeSession |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  |  | No miscolored data present |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  |  | Good: TransmissionSessionStart.indication |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  |  | No RedPartReception.indication |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  |  | NO TransmissionSessionCompletion.indication |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  |  | Good: TransmissionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  |  | Good: ReceptionSessionCancellation.indication |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  |  | Good: InitialTransmissionCompletion.indication |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | NO ReceptionSessionStart.indication |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  |  | No SDA |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  |  | No SDA |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  |  | redPart state diagram |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  |  | greenPart state diagram |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  |  | use IANA |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  |  | No AUTH -- RFC5327 |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  |  | No AUTH -- may do auth |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  |  | No AUTH -- ciphersuite |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  |  | No AUTH -- cookie check |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  |  | No AUTH -- MIB policy check |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  |  | No AUTH -- policy obeyed on receipt check |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  |  | No AUTH -- all or none check. |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  |  | No AUTH -- Auth extension structure |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  |  | Good: UDP is used over loopback to the gateways. |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  |  | UDPPort -- exception |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  |  | Good: SessionNumber OK (65) |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  |  | Good: InitialCheckpointSerialNumber OK (9096) |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  |  | Good: InitialCheckpointSerialNumber: 9096 |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  |  | No UNKNOWN extensions found |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  |  | No CCSDS Space Packet |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  |  | No CCSDS Space Packet |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  |  | Good: CCSDS ENCAP |

# Test: RXMTCYCEXC\_R

Retransmission cycles exceeded at receiver (retransmitting reports)

## Settings

|  |  |  |
| --- | --- | --- |
| Setting | Value | Units |
| LATENCY | 500 | (ms) |
| LOSS | 50 | (plr) |
| RED\_SIZE | 40000 | (Bytes) |
| GREEN\_SIZE | 0 | (Bytes) |
| RATE | 20000 | (Bytes/sec) |
| SEG\_SIZE | 500 | (Bytes) |

## Test Procedure

Turn the error rate up, RedPart size up, and segment size down and then jack with the MAXRXMTCYC value for the receiver (set way below that of the transmitter

## ION to Python Results:

### Sender Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.2 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.1 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

./startIONLTPSender -d 18 -c 6 -s file:redAndGreenParts

while true; do

sleep 5

done

### Receiver Commands

#!/bin/bash

Python\_Files/supportFiles/udp\_encap-over-udp\_gateway.py --local-lan-addr=127.0.0.1 --local-lan-port=2113 \

--remote-lan-addr=127.0.0.1 --remote-lan-port=1113 \

--local-wan-addr=10.0.0.1 --local-wan-port=5555 \

--remote-wan-addr=10.0.0.2 --remote-wan-port=5555 \

--protocol=encapsulation &

sleep 1

Python\_Files/pyltprecv.py -E 18 \

--maxRxmtCycRecv 3 \

-n "{16: {'addr': ('127.0.0.1', 2113), 'XMIT': True, 'RECV': True}}" &

while true; do

sleep 5

done

### Host Commands

None.

### Sender Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(0, 1190), (1191, 2380), (2381, 3570), (3571, 4760), (4761, 5950), (5951, 7140), (7141, 8330), (8331, 9520), (9521, 10710), (10711, 11900), (11901, 13090), (13091, 14280), (14281, 15470), (15471, 16660), (16661, 17849), (17850, 19038), (19039, 20227), (20228, 21416), (21417, 22605), (22606, 23794), (23795, 24984), (24985, 26174), (26175, 27364), (27365, 28554), (28555, 29744), (29745, 30934), (30935, 32124), (32125, 33314), (33315, 34504), (34505, 35694), (35695, 36884), (36885, 38074), (38075, 39264), (39265, 39999), (39265, 39999), (39265, 39999), (0, 1191), (1192, 2380), (5951, 7140), (8331, 9520), (10711, 11900), (20228, 21416), (23795, 24984), (24985, 26174), (26175, 27364), (29745, 30934), (33315, 34504), (36885, 38073), (38074, 38074), (38074, 38074), (38074, 38074), (38074, 38074), (38074, 38074), (38074, 38074), (1192, 2380), (20228, 21416), (23795, 24984), (24985, 26174), (36885, 38072), (38073, 38073), (20228, 21416), (23795, 24984), (24985, 26173), (26174, 26174), (23795, 24984), (24985, 26172)] |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 39999 |
| EORP | 39999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | True |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [] |
| InitialCheckpointSerialNumber | 1703 |
| InitialReportSerialNumber | 214 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1703, 0), (1703, 0), (1703, 0), (1704, 214), (1704, 214), (1704, 214), (1704, 214), (1704, 214), (1704, 214), (1705, 215), (1706, 216)] |
| allReports | [(214, 1703), (215, 1704), (216, 1705), (217, 1706)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 3L, 3L, 8L, 9L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 1L, 1L, 1L, 1L, 1L, 8L, 9L, 0L, 0L, 0L, 0L, 0L, 1L, 8L, 9L, 0L, 0L, 0L, 1L, 8L, 9L, 0L, 0L, 14L, 15L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L] |
| cancelReasonCodes | [5, 5, 5, 5, 5, 5, 5, 5, 5] |
| checkpoints | [(1703, 0), (1704, 214), (1705, 215), (1706, 216)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 26172 |
| highestRedByteIndex | 39999 |
| lastCancelOrAck | 14 |
| lastReportOrAck | ('Ack', 217) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [214, 215, 216, 217] |
| reports | [(214, 1703), (215, 1704), (216, 1705), (217, 1706)] |
| sdnvSizes | [1, 2, 3] |
| segmentTypeCounts | {0: 55, 1: 8, 2: 0, 3: 3, 4: 0, 5: 0, 6: 0, 7: 0, 8: 4, 9: 4, 10: 0, 11: 0, 12: 0, 13: 0, 14: 9, 15: 1} |
| segmentTypes | [0L, 1L, 3L, 8L, 9L, 14L, 15L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Sender Application Output

#### ion\_ltp.out

ltpstart doing options...

ltpstart using ION rc file:, node16.ionrc

[i] admin pgm using default SDR parms.

wmKey: 0

wmSize: 5000000

wmAddress: 0

sdrName: ''

sdrWmSize: 0

configFlags: 13

heapWords: 250000

heapKey: -1

pathName: '/tmp'

Stopping ionadmin.

No security; using empty ionsecrc file node16.ionsecrc\_empty

Stopping ionsecadmin.

Stopping ltpadmin.

eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeefgggggggggggggggggggggggggggggggggg=g=gseeeeeeeeeeeee@gggggggggggggg=g=g=g=g=gseeeeee@gggggggseeee@gggggseee@gggs]gssssssss

### Receiver Summary:

|  |  |
| --- | --- |
|  |  |
| AllGreenDataSegments | [] |
| AllRedDataSegments | [(2381, 3570), (3571, 4760), (4761, 5950), (7141, 8330), (9521, 10710), (11901, 13090), (13091, 14280), (14281, 15470), (15471, 16660), (16661, 17849), (17850, 19038), (19039, 20227), (21417, 22605), (22606, 23794), (27365, 28554), (28555, 29744), (30935, 32124), (32125, 33314), (34505, 35694), (35695, 36884), (38075, 39264), (39265, 39999), (0, 1191), (5951, 7140), (8331, 9520), (10711, 11900), (26175, 27364), (29745, 30934), (33315, 34504), (38074, 38074), (1192, 2380), (36885, 38072), (38073, 38073), (20228, 21416), (26174, 26174), (23795, 24984), (24985, 26172)] |
| AmountGreenDataReceivedByApp | 0 |
| DataBeyondEOB | False |
| DuplicateCancelAck | False |
| DuplicateReportAck | False |
| EOB | 39999 |
| EORP | 39999 |
| GreenSegmentsReceived | [] |
| HasReceivedCancelAck | False |
| IllegalSegments | {} |
| IllegalSegmentsAfterCancel | [0L, 0L] |
| InitialCheckpointSerialNumber | 1703 |
| InitialReportSerialNumber | 214 |
| MiscoloredData | False |
| ReceiveIndications | ['ReceiveSessionStart', 'ReceptionSessionCancellation'] |
| TransmitIndications | ['SessionStart', 'InitialTransmissionCompletion', 'TransmissionSessionCanceled'] |
| UnacknowledgedCheckpoints | [] |
| UnacknowledgedReports | [] |
| allCheckpoints | [(1703, 0), (1704, 214), (1705, 215), (1706, 216)] |
| allReports | [(214, 1703), (215, 1704), (215, 1704), (216, 1705), (217, 1706)] |
| allSegmentsInOrder | [0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 3L, 8L, 9L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 1L, 8L, 8L, 9L, 0L, 0L, 1L, 8L, 9L, 0L, 1L, 8L, 9L, 14L, 0L, 0L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L, 14L] |
| cancelReasonCodes | [5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5] |
| checkpoints | [(1703, 0), (1704, 214), (1705, 215), (1706, 216)] |
| clientServiceID | 6 |
| comments | [] |
| containsGreenData | False |
| containsRedData | True |
| encapsulation | ENCAPSULATION |
| failures | [] |
| headerExtensions | [] |
| highestByteIndex | 26172 |
| highestRedByteIndex | 39999 |
| lastCancelOrAck | 14 |
| lastReportOrAck | ('Ack', 217) |
| lowestGreenByteIndex | False |
| maxHeaderExtensions | 0 |
| maxTrailerExtensions | 0 |
| receiverCancelled | True |
| reportAcks | [214, 215, 216, 217] |
| reports | [(214, 1703), (215, 1704), (216, 1705), (217, 1706)] |
| sdnvSizes | [1, 2, 3] |
| segmentTypeCounts | {0: 33, 1: 3, 2: 0, 3: 1, 4: 0, 5: 0, 6: 0, 7: 0, 8: 5, 9: 4, 10: 0, 11: 0, 12: 0, 13: 0, 14: 11, 15: 0} |
| segmentTypes | [0L, 1L, 3L, 8L, 9L, 14L] |
| segmentVersions | [0L] |
| senderCancelled | False |
| sessionNumber | 1 |
| sessionOriginator | 16 |
| trailerExtensions | [] |
| warnings | [] |

### Receiver Application Output

#### log\_pyltprecv

\*\*\* 2014-10-27 14:09:54,275 - pyltprecv - INFO - pyltprecv LTP engine 18 listening on ('', 1113) client Service 6

\*\*\* 2014-10-27 14:10:04,697 - pyltprecv - INFO - ReceptionSessionStart.indication New session started (16, 1)

\*\*\* 2014-10-27 14:11:35,753 - pyltprecv - INFO - ReceptionSessionCancellation.indication: LTP engine reports cancellation of my session: (16, 1) reason RXMTCYCEXC

\*\*\* 2014-10-27 14:11:35,754 - pyltprecv - INFO - Cancellation included extents:

\*\*\* 2014-10-27 14:11:35,754 - pyltprecv - INFO - (0, 23795) (26174, 13826)

### Sender PCAP

14:10:02.208838 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1191)

14:10:03.191121 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1191, 1190)

14:10:04.173305 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2381, 1190)

14:10:05.155488 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3571, 1190)

14:10:06.137659 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4761, 1190)

14:10:07.119873 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5951, 1190)

14:10:08.102051 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7141, 1190)

14:10:09.084255 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8331, 1190)

14:10:10.066467 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(9521, 1190)

14:10:11.048584 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(10711, 1190)

14:10:12.030535 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(11901, 1190)

14:10:13.012484 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(13091, 1190)

14:10:13.994440 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(14281, 1190)

14:10:14.976435 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(15471, 1190)

14:10:15.958523 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(16661, 1189)

14:10:16.940378 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(17850, 1189)

14:10:17.922365 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(19039, 1189)

14:10:18.904325 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(20228, 1189)

14:10:19.886311 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(21417, 1189)

14:10:20.868314 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(22606, 1189)

14:10:21.850267 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(23795, 1190)

14:10:22.833250 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(24985, 1190)

14:10:23.815936 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(26175, 1190)

14:10:24.798703 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(27365, 1190)

14:10:25.781471 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(28555, 1190)

14:10:26.764296 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(29745, 1190)

14:10:27.747155 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(30935, 1190)

14:10:28.729947 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(32125, 1190)

14:10:29.712608 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(33315, 1190)

14:10:30.695419 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(34505, 1190)

14:10:31.678204 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(35695, 1190)

14:10:32.660984 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(36885, 1190)

14:10:33.643728 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(38075, 1190)

14:10:34.626602 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(39265, 735) CP(1703)

14:10:39.210448 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(39265, 735) CP(1703)

14:10:44.211285 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(39265, 735) CP(1703)

14:10:45.263236 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1703, 214) [0 , 40000] / Raw

14:10:45.266406 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(214)

14:10:45.294034 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:10:46.276794 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1189)

14:10:47.258018 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5951, 1190)

14:10:48.239932 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8331, 1190)

14:10:49.222054 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(10711, 1190)

14:10:50.203892 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(20228, 1189)

14:10:51.185857 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(23795, 1190)

14:10:52.168714 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(24985, 1190)

14:10:53.151398 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(26175, 1190)

14:10:54.134234 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(29745, 1190)

14:10:55.117023 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(33315, 1190)

14:10:56.099740 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(36885, 1189)

14:10:57.081717 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38074, 1) CP(1704)

14:11:03.214646 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38074, 1) CP(1704)

14:11:07.215339 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38074, 1) CP(1704)

14:11:12.216381 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38074, 1) CP(1704)

14:11:17.217328 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38074, 1) CP(1704)

14:11:22.218406 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38074, 1) CP(1704)

14:11:25.239176 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1704, 215) [0 , 40000] / Raw

14:11:25.241627 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(215)

14:11:25.269105 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1189)

14:11:26.250333 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(20228, 1189)

14:11:27.235131 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(23795, 1190)

14:11:28.217823 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(24985, 1190)

14:11:29.200511 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(36885, 1188)

14:11:30.181663 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38073, 1) CP(1705)

14:11:31.217273 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1705, 216) [0 , 40000]

14:11:31.219620 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(216)

14:11:31.246834 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(20228, 1189)

14:11:32.228686 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(23795, 1190)

14:11:33.211429 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(24985, 1189)

14:11:34.193394 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(26174, 1) CP(1706)

14:11:35.234653 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1706, 217) [0 , 40000]

14:11:35.236741 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(217)

14:11:35.264426 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(23795, 1190)

14:11:36.247242 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(24985, 1188)

14:11:36.257842 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:37.228321 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xF Cancel-acknowledgment segment to block receiver

14:11:40.259711 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:42.261060 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:44.262669 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:46.264091 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:48.265938 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:52.269212 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:54.272517 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:56.274092 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

### Receiver PCAP

14:10:04.673398 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(2381, 1190)

14:10:05.655548 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(3571, 1190)

14:10:06.637747 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(4761, 1190)

14:10:08.602121 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(7141, 1190)

14:10:10.566540 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(9521, 1190)

14:10:12.530610 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(11901, 1190)

14:10:13.512552 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(13091, 1190)

14:10:14.494513 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(14281, 1190)

14:10:15.476485 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(15471, 1190)

14:10:16.458586 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(16661, 1189)

14:10:17.440444 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(17850, 1189)

14:10:18.422429 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(19039, 1189)

14:10:20.386368 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(21417, 1189)

14:10:21.368365 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(22606, 1189)

14:10:25.298747 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(27365, 1190)

14:10:26.281537 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(28555, 1190)

14:10:28.247225 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(30935, 1190)

14:10:29.230005 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(32125, 1190)

14:10:31.195451 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(34505, 1190)

14:10:32.178245 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(35695, 1190)

14:10:34.143765 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(38075, 1190)

14:10:44.711344 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x3 Red data, Checkpoint, EORP, EOB C(6) Data(39265, 735) CP(1703)

14:10:44.763185 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1703, 214) [0 , 40000] / Raw

14:10:45.766463 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(214)

14:10:45.794090 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(0, 1192)

14:10:47.758085 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(5951, 1190)

14:10:48.740005 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(8331, 1190)

14:10:49.722125 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(10711, 1190)

14:10:53.651477 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(26175, 1190)

14:10:54.634433 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(29745, 1190)

14:10:55.617151 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(33315, 1190)

14:11:22.718471 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38074, 1) CP(1704)

14:11:22.737418 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1704, 215) [0 , 40000] / Raw

14:11:24.739109 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1704, 215) [0 , 40000] / Raw

14:11:25.741693 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(215)

14:11:25.769138 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(1192, 1189)

14:11:29.700570 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(36885, 1188)

14:11:30.681705 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(38073, 1) CP(1705)

14:11:30.717230 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1705, 216) [0 , 40000]

14:11:31.719654 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(216)

14:11:31.746957 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(20228, 1189)

14:11:34.693458 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x1 Red data, Checkpoint, NOT (EORP or EOB) C(6) Data(26174, 1) CP(1706)

14:11:34.734602 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x8 Report segment RS(1706, 217) [0 , 40000]

14:11:35.736824 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x9 Report-acknowledgment segment RA(217)

14:11:35.757797 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:35.764487 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(23795, 1190)

14:11:36.747323 Ether / IP / UDP 10.0.0.2:5555 > 10.0.0.1:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0x0 Red data, NOT (Checkpoint, EORP or EOB) C(6) Data(24985, 1188)

14:11:37.758299 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:39.759644 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:41.760997 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:43.762603 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:45.764014 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:47.765876 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:49.767676 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:51.769150 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:53.772449 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

14:11:55.774008 Ether / IP / UDP 10.0.0.1:5555 > 10.0.0.2:5555 / CCSDSENCAP /   
LTPv0L SID(16, 1) 0xE Cancel segment from block receiver Reason(RXMTCYCEXC)

## There Are No Python to ION Results:

There are no results for this test.

## Comments:

### Python to ION Comments

Hard to elicit this behavior from the ION implementation. The Python implementation allows the retransmission cycle limit to be set from the command line.

## Comparison of Test Results and Requirements

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section 6 Reference | LTP for CCSDS Requirement ID | Requirement Text | Extra Success Criteria. | Look For | ION->Python | Python->ION |
| 6.2.1.1 | RFC5326 3.0 | Segment structure -- Each LTP segment comprises: a "header" in the format defined below; zero or more octets of "content"; zero or more octets of "trailer" as indicated by information in the "Extensions field" of the header. |  |  | Good: Segment Types: 0 , 1 , 3 , 8 , 9 , 14 |  |
| 6.2.1.1 | RFC5326 3.1(.0) | Segment Header consists of 1) control byte 2) session ID 3) Extensions. Control Byte contains segment type. | Need to cover segment types: 0--4, 7--9, 12--15 (rest are undefined) |  | Good: Segment Types: 0 , 1 , 3 , 8 , 9 , 14 |  |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Field Format (in header) -- The Extensions field enables the inclusion of zero or more functional extensions to the basic LTP segment, each in type-length-value (TLV) |  |  | No Extensions |  |
| 6.2.1.1 | RFC5326 3.1.4 | Extensions Format -- Extensions are formed by a 1 octet id followed by SDNV length then value. [header / trailer] |  |  | No Extensions |  |
| 6.2.1.1 | RFC5326 3.2.1 | Data Segment Format -- Data Segments contain client service ID, offset, length, checkpoint serial number (if checkpoint) report serail number (if checkpoint queued in response to a report, otherwise zero), and client service data. |  |  | Good: CSID Offset Length CSN CSN\_RSN Data |  |
| 6.2.1.1 | RFC5326 3.2.2 | Report Segment Format -- RSs have a report serial number, checkpoint serial number, upper bound, lower bound, reception claim count, reception claims [offset, length]; reception claims are always of length >0 and stay inside bounds of the report |  |  | Good: RSN RSN\_CSN [(214, 1703), (215, 1704), (215, 1704), (216, 1705), (217, 1706)] |  |
| 6.2.1.1 | RFC5326 3.2.3 | Report-Acknowledgement Segment Format -- RA segments contain a report serial number. |  |  | Good: Report Acknowledgement(s) sent: [214, 215, 216, 217] |  |
| 6.2.1.1 | RFC5326 3.2.4 | Cancel segment reason codes | Need to cover: 0 (USR) 1 (UNREACH) 2 (RLEXC) 3 (MISCOLORED) 4 (SYS\_CNCLD) 5 (RXMTCYCEXC) | Ensure: CancelReasonCodes: 0x05 | Good: CancelReasonCodes: 0x05 |  |
| 6.2.1.1 | RFC5326 3.3 | Segment Trailer contains extnsions encoded as TLVs |  |  | No Trailers |  |
| 6.2.1.1 | RFC5326 4.1 | Transmission.request primitive |  |  | Good: TransmissionRequest.request |  |
| 6.2.1.1 | RFC5326 4.2 | Cancellation.request primitive |  | No cancellation request primitive attempted | No cancellation request primitive attempted |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 1 -- Whenever the content of any of the fields of the header of any received LTP segment does not conform to this specification document, the segment is assumed to be corrupt and MUST be discarded immediately and processed no further |  |  | No corrupt segments. |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 2 -- if no client service id exists at local LTP engine and no transmission queue to segment sender, silently drop segment |  |  | No check -- receive client service exists. |  |
| 6.2.1.1 | RFC5326 6.0 | Overriding Rule 3 -- if no client service id exists at local LTP engine but queue to segment sender send UNREACH |  |  | No check -- receive client service exists. |  |
| 6.2.1.1 | RFC5326 6.1 | Start Transmission -- Receipt of link cue that transmission can begin results in starting transmission. |  |  | Good: StartTransmission |  |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- When a checkpoint is transmitted, start checkpoint timer. |  |  | Good: Checkpoint transmitted: [(1703, 0), (1703, 0), (1703, 0), (1704, 214), (1704, 214), (1704, 214), (1704, 214), (1704, 214), (1704, 214), (1705, 215), (1706, 216)] |  |
| 6.2.1.1 | RFC5326 6.2 | Start Checkpoint Timer -- Immediately suspend timer if it is known that the receiver is not transmitting. |  |  | No check: receiver is transmitting. |  |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- When a report segment is transmitted, start the report segment timer. |  |  | Good: StartReportSegment Timer (Report(s) retransmitted) |  |
| 6.2.1.1 | RFC5326 6.3 | Start Report Segment Timer -- Immediately suspend timer if it is known that the remote engine is not transmitting. |  |  | No check -- receiver is transmitting |  |
| 6.2.1.1 | RFC5326 6.4 | Stop Transmission -- On receipt of a link cue that transmission should be stopped, stop transmitting. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.5 | Suspend Timers -- When transmission is halted, stop countdown timers associated with the remote engine. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.6 | Resume Timers -- When transmission is resumed, resume countdown timers associated with the remote engine. |  |  | No Link Cues |  |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has NOT expired, retransmit the checkpoint. |  |  | Good: checkpoints retransmitted: [(1704, 6, 215), (1705, 1, 216), (1706, 1, 217), (1703, 3, 214)] |  |
| 6.2.1.1 | RFC5326 6.7 | Retransmit Checkpoint -- When a checkpoint timer expires: If the retransmission limit has expired, cancel the session. |  |  | No RXMTCYCEXC detected from LTP checkpoint sender. |  |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has NOT expired, retransmit the report |  |  | Good: Report retransmitted |  |
| 6.2.1.1 | RFC5326 6.8 | Retransmit Report Segment -- When a report timer expires: If the retransmission count has expired, cancel the session with reason RLEXC. |  |  | No cancel due to RLEXC |  |
| 6.2.1.1 | RFC5326 6.9 | RedPartReception.indication |  |  | No RedPartReception.indication |  |
| 6.2.1.1 | RFC5326 6.10 | GreenPartSegmentArrival.indication |  |  | No GreenPartReception.indication |  |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems does NOT exceed limit, send a report in response to a checkpoint; note formatting and semantic rules of 6.11. |  |  | Good: SendReceptionReport |  |
| 6.2.1.1 | RFC5326 6.11 | Send Reception Report -- if the number of reception problems exceeds limit, cancel the session with reason RLEXC. |  |  | Good: RLEXC due to report retransmission. |  |
| 6.2.1.1 | RFC5326 6.12 | InitialTransmissionCompletion.indication -- provide this indication to the sending client when all data transmitted and all red data (if any) known received. |  |  | Good: InitialTransmissionCompletion.indication |  |
| 6.2.1.1 | RFC5326 6.13 | Retransmit Data -- triggered by receipt of a report segment; send Report Ack; retransmit data as indicated by report. |  |  | No red data retransmission |  |
| 6.2.1.1 | RFC5326 6.14 | Stop Report Segment (RS) Timer -- stop RS timer on receipt of a report ack for the report. |  |  | Good: no report with same RSN sent after report ack for that RSN. |  |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- when a cancel semgnet is sent, start cancel timer. |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.15 | Start Cancel Timer -- Immediately suspend if known that the peer is not transmitting. |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions exceeds the limit, cancel the session. |  |  | No session cancellation due to too many cancel segment retransmissions |  |
| 6.2.1.1 | RFC5326 6.16 | Retransmit Cancellation Segment -- If the number of retransmissions does NOT exceed the limit, retransmit the cancel segment (with the same reason-code as before). |  |  | Good: Cancel segment retransmission |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation -- In response to receipt of a cancel segment: If there is no transmission queue to the receiver, no action. |  |  | No check: both sides have transmission queues. |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Sender (CS) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancel segment, send a cancel acknowledgment to block sender (CAS) segment. If there IS a record of the session, cancel session. |  |  | NO cancelACK to sender |  |
| 6.2.1.1 | RFC5326 6.17 | Acknowledge Cancellation to Recevier (CR) -- in response to receipt of a cancel segment: If there is a transmission queue to the sender of the cancellation segment, send a cancel acknowledgement to block receiver(CAR) segment. If there IS a record of the session, cancel session. |  |  | No cancelACK to receiver |  |
| 6.2.1.1 | RFC5326 6.18 | Stop Cancel Timer -- in response to receipt of a cancellation acknowledgement, stop the cancel timer. |  |  | No check: receiver blocked from receiving cancel ACK |  |
| 6.2.1.1 | RFC5326 6.19 | Cancel Session: dump everything queued for transmission; stop all countdown timers. |  |  | Good: Sender gives up after receiving cancel from receiver. |  |
| 6.2.1.1 | RFC5326 6.20 | Close Session -- on session closure, stop any countdown timers and remove session (session no longer recognized) |  |  | CHECKME: closeSession |  |
| 6.2.1.1 | RFC5326 6.21 | Handle Miscolored Segment -- discard segment and cancel session. |  |  | No miscolored data present |  |
| 6.2.1.1 | RFC5326 7.1 | TransmissionSessionStart.indication |  |  | Good: TransmissionSessionStart.indication |  |
| 6.2.1.1 | RFC5326 7.3 | Red-Part Reception.indication |  |  | No RedPartReception.indication |  |
| 6.2.1.1 | RFC5326 7.4 | Transmission-Session Completion.indication |  |  | NO TransmissionSessionCompletion.indication |  |
| 6.2.1.1 | RFC5326 7.5 | Transmission-Session Cancellation.indication |  |  | Good: TransmissionSessionCancellation.indication |  |
| 6.2.1.1 | RFC5326 7.6 | Reception-Session Cancellation.indication |  |  | Good: ReceptionSessionCancellation.indication |  |
| 6.2.1.1 | RFC5326 7.7 | Initial-Transmission Completion.indication |  |  | Good: InitialTransmissionCompletion.indication |  |
| 6.2.1.12 | RFC5326 7.1 | ReceptionSessionStart.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | Good: ReceptionSessionStart.indication |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: RedPartReception.indication -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: Transmission.request -- LTP SDA shall present to clients the Transmission.request, InitialTransmissionCompletion.indication and RedPartReception.indication primitives that are defined in the LTP service specification. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.1.1 | SDA: InitialTransmissionCompletion.indication Note: Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.1 | SDA Client Service ID is '2' -- The client service ID passed by SDA to LTP shall be '2', signifying LTP Service Data Aggregation. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.2.2 | SDA: Entire SDA SDU passed to LTP is red -- The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.4.1.2 | SDA: Size limit reached -- If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data. The SDA client data capsules included in the service data unit for the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.4.1.3 | SDA: Time limit reached -- When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a Transmission.request primitive to the LTP engine.  The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.  The Transmission.request presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.  The SDA client data capsules included in the Transmission.request to LTP shall be removed from the SDA set of retained client data. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.4.3 | SDA: TransmissionSessionCompletion.indications -- Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.5.1 | SDA: Red-Part reception, extraction, and delivery -- Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter. |  |  | No SDA |  |
| 6.2.1.12 | CCSDS 7.2.2.5.2 | SDA: Reception cancellation.indications -- Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID. |  |  | No SDA |  |
| 6.2.1.1 | 8.1 (red) | State Transition diagram |  |  | redPart state diagram |  |
| 6.2.1.1 | 8.2 (green) | State Transition diagram |  |  | greenPart state diagram |  |
| 6.2.1.12 | 10.2 | Use IANA registry for LTP Extension identifiers |  |  | use IANA |  |
| 6.2.1.2 | CCSDS 3.1.2 | LTP Authentication per RFC5327 -- This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document. |  |  | No AUTH -- RFC5327 |  |
| 6.2.1.2 | CCSDS 3.9.1 | LTP Authentication -- Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327. |  |  | No AUTH -- may do auth |  |
| 6.2.1.2 | CCSDS 3.9.4 | Ciphersuite Identification -- If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]). |  |  | No AUTH -- ciphersuite |  |
| 6.2.1.3 | CCSDS 3.9.2 | No Cookies -- Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327. |  |  | No AUTH -- cookie check |  |
| 6.2.1.2 | CCSDS 3.9.3 | MIB Control of Authentication -- If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both. |  |  | No AUTH -- MIB policy check |  |
| 6.2.1.2 | CCSDS 3.9.5 | MIB Control of Authentication --If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer. |  |  | No AUTH -- policy obeyed on receipt check |  |
| 6.2.1.2 | CCSDS 3.9.6 | Authentication is 'all-or-none' -- If authentication is used, it must be included on either all LTP segments [of a session] or none. |  |  | No AUTH -- all or none check. |  |
| 6.2.1.2 | RFC5327 2.1 | Authentication Structure -- LTP Authentication extension structure and ciphersuite definitions. |  |  | No AUTH -- Auth extension structure |  |
| 6.2.1.4 | CCSDS 3.3.1 | LTP over UDP -- Contrary to section 5 of RFC5326, this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks. |  |  | Good: UDP is used over loopback to the gateways. |  |
| 6.2.1.4 | CCSDS 3.3.2 | UDP Port 1113 -- Implementations of LTP over UDP should use the 'ltp-deepspace' UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]) |  |  | UDPPort -- exception |  |
| 6.2.1.5 | CCSDS 3.5.1 | Session Number Selection -- Session numbers chosen by sending LTP engines must be in the range [1, 232-1] |  |  | Good: SessionNumber OK (1) |  |
| 6.2.1.6 | CCSDS 3.5.2 | Initial Checkpoint Serial Number Range -- The initial checkpoint serial number values used by conformant implementations must be in the range [1, 214-1]. |  |  | Good: InitialCheckpointSerialNumber OK (1703) |  |
| 6.2.1.6 | CCSDS 3.5.3 | Initial Checkpoint Serial Number Value -- The initial checkpoint serial number values used by conformant implementations should be chosen at random. |  |  | Good: InitialCheckpointSerialNumber: 1703 |  |
| 6.2.1.7 | CCSDS 3.8.1 | Ignore unknown LTP Extensions -- Implementations must ignore LTP extensions that they do not know how to process on receipt. |  |  | No UNKNOWN extensions found |  |
| 6.2.1.9 | CCSDS A3.1 | LTP Encapsulation in Space Packets -- When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service. |  |  | No CCSDS Space Packet |  |
| 6.2.1.9 | CCSDS A3.2 | LTP in Space Packets APID -- The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10]) |  |  | No CCSDS Space Packet |  |
| 6.2.1.8 | CCSDS A2.1 | Encapsulation in Encap Packets -- When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service. |  |  | Good: CCSDS ENCAP |  |