

|  |
| --- |
| CCSDS Bundle Protocol Interoperability Testing |

DRAFT CCSDS Record

CCSDS 000.0-Y-0

Draft Yellow Book

May 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 | CCSDS BUNDLE PROTOCOL INTEROPERABILITY TESTING Draft CCSDS Record, Issue 0 | May 2015 | Current draft |
|  |  |  |  |
|  |  |  |  |

CONTENTS

Section Page

[DOCUMENT CONTROL iii](#_Toc418256020)

[CONTENTS iv](#_Toc418256021)

[1 Introduction 1-1](#_Toc418256022)

[1.1 Purpose 1-1](#_Toc418256023)

[1.2 Scope 1-1](#_Toc418256024)

[1.3 Applicability 1-1](#_Toc418256025)

[1.4 References 1-1](#_Toc418256026)

[2 Overview 2-3](#_Toc418256027)

[2.1 CCSDS Interoperability Testing 2-3](#_Toc418256028)

[2.2 DTPC Interoperability Testing 2-3](#_Toc418256029)

# Introduction

## Purpose

This report documents the interoperability testing conducted in support of the CCSDS Bundle Protocol 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.

The two implementations used for these tests were:

* The BP implementation that comes with the ION 3.2.2 open-source distribution.
* The BP implementation that comes with the SourceForge DTN2.9 open-source distribution and modified to support ISS operations.

## Scope

This document applies to the CCSDS Bundle Protocol specification only.

## Applicability

This document supports the interoperability testing requirement needed for the CCSDS Bundle Protocol 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] CCSDS Bundle Protocol. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 734.2-R-1. Washington, D.C.: CCSDS, XXXXX 2014

[X] 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

[X] RFC6260

[X] UDP

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

# Overview

NOTE: Everything tested in at least one direction except those capabilities that don’t bear on interoperability.

## CCSDS Interoperability Testing

Test participants: MSFC, JAXA

Test Plan

* Attachment: excel sheet
* Addendum: David’s final interoperability testing update email

## DTPC Interoperability Testing

Executed by TopCoder

1. : CCSDS Interoperability Testing

NOTE – This Annex is a slightly-edited version of the document ‘TP JAXA-NASA Joint Test Plan v11 7.docx’

* 1. Organization

Section A2 lists the requirements from the PICS section of the BP-for-CCSDS document. Section A4 lists the set of tests that were performed to exercise the two implementations and to test each item in the requirements list.

* + 1. Test Configuration
* Series of 4 nodes alternating DTN2 and ION; ION A, DTN2 B, ION C, DTN2 D.
* Unique test stations at each end based on test conductors particular setup
* Two test conductors; one at MSFC and one at Tsukuba
* Connections are protected by IPSEC VPN
* Checkpoint gateway at MSFC
* OpenSwan gateway at Tsukuba

****

* + 1. Software Configuration

1. DTN2.9.x
   1. Latest items which support ISS
   2. HOSC DTN2 implementation based on DTN2.9
   3. Executing on SuperMicro/KVM virtual servers
   4. Linux based (kernel-2.6.18/gcc-4.1.2)
   5. Test tool kit
      1. DTNperf\_v3.3.3
      2. dtnping
      3. dtnsink
      4. dtnsend
2. ION 3.2.2
   1. No Special patches
   2. Using publically available ION distribution on Sourceforge
   3. Based on 3.2.2
   4. Executing on Intel VT/KVM virtual servers
   5. Linux based (kernel-2.6.32/gcc-4.4.6)
   6. Test tool kit
      1. DTNperf\_v3.3.2
      2. bping
      3. bpsink
      4. bptrace
      5. bpsource
      6. Protocol Inclusions
3. LTP 734-1-R-3, Licklider Transmission Protocol Specification
4. ECOS 734.2-R-3, CCSDS Bundle Protocol Specification
5. ACS 734.2-R-3, CCSDS Bundle Protocol Specification.
6. DTPC 734.2-R-3, CCSDS Bundle Protocol Specification.
7. CBHE RFC 6260
8. UDP RFC 5405
   * 1. IPN IDs

The selection of IPNs is made to ensure that CBHE is exercised and that earlier deficiencies in applications have not been propagated forward into the current code base. Other nodes may be used to inject test scenarios into the network but must be coordinated prior to testing.

1. Nodes A and C; 17000, 20000
2. Nodes B and D; 19000, 21000
   1. BP for CCSDS Requirements

Section 6 of the BP for CCSDS Book 734-2-r-3 describes the Protocol Implementation Conformance requirements on BP for CCSDS implementations. The requirements are listed in the following table. In most cases where a PICS requirement references several requirements in RFC5050, the individual requirements have been broken out to facilitate mapping of tests to requirements. In these cases, the PICS requirement line is listed in gray/brown with the individual RFC5050 requirements listed in separate white cells underneath. Annex requirements are listed separately.

| **PICS Requirement** | **BP for CCSDS Requirement** |
| --- | --- |
| 6.2.1.1 | All Bundle Protocol implementations shall implement the following capabilities in accordance with the base standard (RFC5050): |
| 6.2.1.1.a | Bundle structure as described in RFC5050 sections 3.1, 4.0, 4.2, 4.4, and 5.8, 8 |
| 6.2.1.1.b | Block structure as described in RFC5050 sections 4.1, 4.5, 4.5.1, 4.5.2, 4.5.3, 4.6, 4.7 |
| 6.2.1.1.c | Administrative Record generation and structure as described in RFC5050 section 5.1, 6.0, 6.1, and 6.2 |
| 6.2.1.1.d | Administrative record processing as described in RFC5050 sections 6.1.1 and 6.1.2, 6.3 |
| 6.2.1.1.e | CBHE in accordance with RFC 6260 and section 3 of this document |
| 6.2.1.1.f | ECOS in accordance with ANNEX C and section 3 of this document |
| 6.2.1.2 | Bundle Protocol Senders |
| 6.2.1.2.1 | A conforming BP implementation shall support the following in accordance with the base standard: |
| 6.2.1.2.1.a | Bundle transmission as described in RFC5050 sections 3.3, 4.3, 5.15, and 5.2 |
| 6.2.1.2.1.b | Bundle forwarding as defined in RFC5050 sections 4.2 5.1, 5.3, 5.4, 5.4.1, 5.4.2, and 5.5 |
| 6.2.1.2.2 | In addition, a BP sender shall also support the following capabilities in accordance with the base standard: |
| 6.2.1.2.2.a | Intermittent connectivity conditions specified in RFC5050 section 1 |
| 6.2.1.2.2.b | Late binding as described in RFC5050 section 1 |
| 6.2.1.2.2.c | Bundle delivery failure as defined in RFC5050 section 3.1 |
| 6.2.1.2.2.d | Bundle priority as defined in RFC5050 section 4.2. and the ECOS appendix C. |
| 6.2.1.2.2.e | Bundle deletion procedures as defined in RFC5050 sections 3.1, 4.2, 5.13, and 5.14. |
| 6.2.1.2.2.f | Dictionary byte array and revision per RFC5050 sections 4.4 and 4.7. |
| 6.2.1.3 | Bundle Protocol Receivers |
| 6.2.1.3.1 | A conforming BP implementation shall support the following in accordance with the base standard: |
| 6.2.1.3.1.a | Bundle acceptance in accordance with RFC5050 section 4.2, 4.5.1, 4.5.2, 5.6, 5.7, 5.9, 5.10, 5.13 |
| 6.2.1.3.1.b | Processing of custody signals as described in RFC5050 sections 3.1, 4.2, 5.4, 5.4.1, 5.4.2, 5.10.1, 5.10.2, 5.11, 5.12, 6.1, 6.1.2, 6.3 |
| 6.2.1.3.1.c | Node registration as defined in RFC5050 section 3.3 and 5.16 |

|  |  |
| --- | --- |
| B2 | CONVERGENCE LAYER ADAPTERS |
| B2.1 | Compliant implementations shall implement at least one of the CLAs in this section. |
| B2.2 | Convergence Layer Adapters shall support the compressed bundle header encoding mechanisms of RFC6260. |
| B3 | LTP CONVERGENCE LAYER ADAPTER |
| B3.1.1 | LTP usage per sections B3.1.2.1, B3.1.2.2, and B3.1.3 of BP CCSDS Spec |
| B4 | UDP CONVERGENCE LAYER ADAPTER |
| B4.1 | UDP usage per section B4.1.a, B4.1.b, B4.1.c, and B4.1.d |
|  | Bundles larger than the maximum MTU must be fragmented. |

|  |  |
| --- | --- |
| C2 | ECOS Block Format |
|  | ECOS block format shall conform to section 4.5.2 and 4.6 of RFC 5050. |
|  | Block extensions shall conform to sections C2.a, C2.b, C2.c, C2.d, C2.e, C2.f, C2.g, and C2.h of the BP CCSDS Spec. |
| C3 | ECOS Block Procedures |
| C3.1 | Structural constraints are defined by C3.1.1, C3.1.2, C3.1.3, and C3.1.4 |
| C3.2 | ECOS extension will be exercised and must traverse the entire network. |
| C3.3 | Bundle forwarding shall conform to RFC 5050 4.6 and 5.4 |
|  | Bundle forwarding shall conform to section 3.3.2 |
| C4 | Bundle delivery shall conform to RFC 5050 5.7 |
| C6 | ECOS implementations shall utilize an IANA defined Bundle Block Type |

Table A-1: PICS Requirements

Annex E testing of Delay-Tolerant Payload Conditioning is expected to be accomplished by Top Coder testing with support by NASA’s Advanced Exploration Systems. Therefore it is not contained in this test plan.

NOTE – The TopCoder DTPC test results are included in Annex B of this document.

* 1. Test Flow for Generating Bundles

Testing was conducted by JAXA and HOSC test teams via G2G VPN with the coordinated configuration provided by the two teams. The VPN encompassed the test machines composed of two DTN routers at each site connected in a linear fashion. Traffic crossed the VPN three times when doing end-to-end testing. However as shown in Figure A-1, the test configuration allows for configurations supporting 4, 3, and 2 nodes.

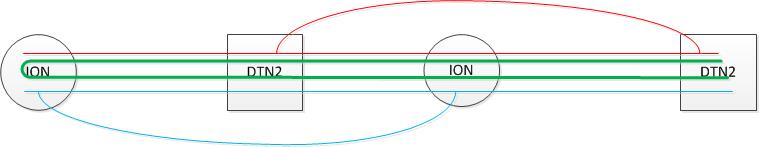


Figure A-1: DTN Test Configurations

Each test case can be initiated from either participant with coordination from the other participant to ensure properly configured routes. Once the VPN was established, the HOSC utilized ICMP Ping to identify and ensure all nodes are online.

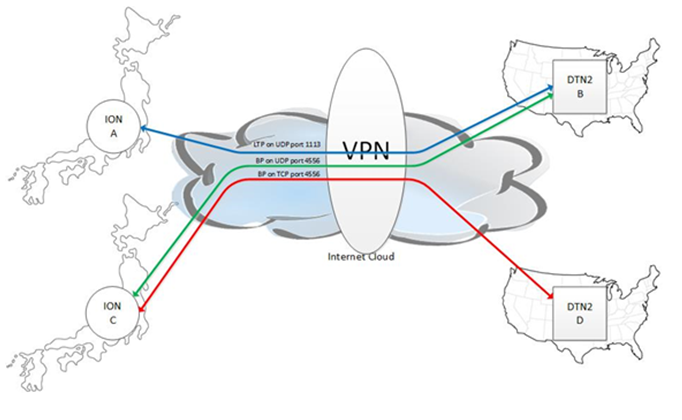


Figure A-2: DTN Test Architecture

Tests exercised the BP protocol with all nodes located at the users’ base of operations. Therefore the DTN2 test team was at MSFC/HOSC/USA and the ION team was at [Tsukuba](http://en.wikipedia.org/wiki/Mount_Tsukuba), Ibaraki, Japan. A single Gateway to Gateway (G2G) VPN was connected the two facilities.

Each facility is responsible to establish a proper test environment with adequate resources. The

G2G VPN was IPSec based and utilize the current HOSC baseline. The HOSC nodes were be privately addressed. The following items applied:

Joint - Create G2G VPN with JAXA

* 1. HOSC - Provide JAXA with documentation stating our VPN requirements for establishing a G2G VPN.
  2. Joint - Network administrator establish G2G VPN to the HOSC Mission Support (UMS) Firewall.
  3. Build rules for G2G VPN and allow the following traffic between JAXA and the assigned UMS DMZ Systems

* + 1. Port 1113 UDP/TCP
    2. Port 4556 UDP/TCP
  1. Joint - Once established, verify connectivity through the use of ICMP Ping.
  2. Recommended tool to transmit and receive data is DTNPerf\_v3.x.
  3. CCSDS Test Cases for BP

There are four basic test cases which exercise basic capabilities. Test cases are forward and reverse in order to exercise source and destination nodes adequately. Figure 5-1 below also lists the requirements satisfied



**Figure A-3: Basic Test Cases**

The basic test cases exercised BP encapsulated within several different protocols to exercise various convergence layers to include LTP, UDP, and TCP. The different protocols were between different nodes of the topology per the test cases of Figure 4-2.

* + 1. Additional Test Cases

Failure cases were used to evaluate routing behavior of BP and demonstrate BP’s resiliency. The test cases are selected based on features of various RFCs which are codified by CCSDS 734-2-r-3. Three general categories exist to bound testing:

1. Failure Cases – Routing behavior
2. Bad Data – Misconfigured bundle
3. Other – Bundle behavior to include ECOS
   * + 1. Failure Test Cases

The Failure test cases are listed in Table 5-1 and exercise various aspects of routing behavior.

| **name** | **Description** | **Direction** | **Results** |
| --- | --- | --- | --- |
| CF0.a | A sends to nonexistent node E but A's routing table causes forwarding to B.  B has no route to E and refuses custody with 'custody failed'. | A>B | Routing behavior |
| CF0.b | D sends to nonexistent node E but D's routing table causes forwarding to C.  C has no route to E and refuses custody with 'custody failed'. | D>C | Routing behavior |
| CF0.c | Node A sends a valid bundle with source EID of dtn:none | A>B>C>D | Routing behavior |
| CF0.d | Node D sends a valid bundle with source EID of dtn:none | D>C>B>A | Routing behavior |
| CF0.e | A sends to D via B and C, C can't get to D and holds onto the bundle until it expires. | A>B>C | Routing behavior |
| CF0.f | D sends to A via C and B, D can't get to C and holds onto the bundle until it expires. | D | Routing behavior |
| CF0.g | C sends to D, block the CLA to D and then cancel the bundle (at C) | C>D | Routing behavior |
| CF0.h | D sends to C, block the CLA to C and then cancel the bundle (at D) | D>C | Routing behavior |

**Table A-2: Failure Test Cases**

* + - 1. Bad Data Test Cases

The Bad Data test cases are listed in Table 5-2 and test the disposition of misconfigured bundles.

| **name** | **Description** | **Direction** | **Results** |
| --- | --- | --- | --- |
| BD0.a | A sends to D via B with a block that B doesn't understand and the BPContol Flag is set to delete the bundle  \*\* Requires a modified version of ION | A>B>C>D | Routing behavior |
| BD0.b | D sends to A via C with a block that C doesn't understand and the BPContol Flag is set to delete the bundle  \*\* Requires a modified version of DTN2 | D>C>B>A | Routing behavior |

**Table A-3: Bad Data Test Cases**

* + - 1. Other Test Cases

The Other test cases are listed in Table 5-3 and exercise LTP and routing behavior to include ECOS.

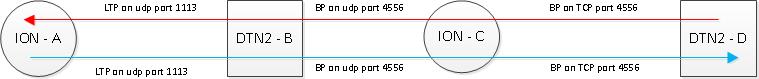
| **name** | **Description** | **Direction** | **Results** |
| --- | --- | --- | --- |
| a | SDA timeout using 4 nodes linear topology (A, B, C, D) A->B LTPCL (vanilla), B->C UDPCL; C->D TCPCL. Send valid bundles A->D with all status report flags set and ***no*** custody transfer. Select bundle sizes to prevent SDA size limit reached. | A>B>C>D | Routing behavior |
| b | SDA timeout using 4 nodes linear topology (A, B, C, D) D->C TCPCL, C->B UDPCL; B->A LTPCL (vanilla). Send valid bundles A->D with all status report flags set and ***no*** custody transfer. Select bundle sizes to prevent SDA size limit reached. | D>C>B>A | Routing behavior |
| c | SDA transmission by re-executing "a" and adding extra bundles to induce transmission prior to timeout | A>B>C>D | Routing behavior |
| d | SDA transmission by re-executing "b" and adding extra bundles to induce transmission prior to timeout | D>C>B>A | Routing behavior |
| e | ECOS testing to demonstrate reordering of bundles sends several bundles A->D via B and C in reverse priority (ECOS) order. Hold transmission at C until all bundles are received then show that they are transmitted in ECOS priority order when released. | A>B>C>D | Routing behavior |
| f | ECOS testing to demonstrate reordering of bundles sends several bundles D->A via C and B in reverse priority (ECOS) order. Hold transmission at B until all bundles are received then show that they are transmitted in ECOS priority order when released. | D>C>B>A | Routing behavior |
| g | ECOS Streaming bit set results in LTP Green transmission from A to B. Send valid bundles A->D with ECOS Streaming (unreliable option) bit set. Use Wireshark to capture and inspect LTP traffic between A and B. | A>B>C>D | Routing behavior |
| h | ECOS Streaming bit set results in LTP Green transmission from B to A. Send valid bundles D->A with ECOS Streaming bit set (unreliable option). Use Wireshark to capture and inspect LTP traffic between B and A. | D>C>B>A | Routing behavior |
| i | ECOS Critical bit set results in transmission over all plausible routes. Send valid bundle A->D with ECOS Critical bit set.   * Configure routes A->B->D and A->C->D   + Both C and B should receive the bundle and forward * Also configure routes B->A and C->A   If A receives the bundle back it should not forward it | A>B>D  A>C>D | Routing behavior |
| j | ECOS Critical bit set results in transmission over all plausible routes. Send valid bundle D->A with ECOS Critical bit set.   * Configure routes D->C->A and D->B->A   + Both C and B should receive the bundle and forward * Also configure routes C->D and B->D   If D receives the bundle back it should not forward it | D>C>A  D>B>A | Routing behavior |

**Table A-4: Other Test Cases**

* 1. Test Procedures
     1. Basic Test Cases Procedures

Four test cases were considered basic and exercised core capabilities to include three Convergence Layer Adaptors: LTP, TCP, and UDP. Fragmentation, Custody Transfer and Aggregated Custody Signals were also exercised.

These test cases are characterized by 4 nodes in a linear topology (A, B, C, D). Figure A-4 illustrates the proposed node configuration with data flows from HOSC and JAXA. Table A-4 details the node configurations for the basic test cases.



**Figure A-4: Basic Test Cases – Data Flow**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 1700017000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 1700017000 ipn:20000.0 |

**Table A-5 Basic Test Cases – Node Configuration**

* + - 1. Basic Test Cases TC0.a and TC0.b

In the first test case (TCO.a), ION – A sent valid bundles A->D with all status report flags set and no custody transfer in the following order; A->B LTPCL; B->C UDPCL; C->D TCPCL. It is recommended that the initial test is with small bundles to eliminate fragmentation. A second iteration utilized larger bundles to ensure fragmentation.

In the second test case (TCO.b), DTN2 – D sent valid bundles D->A with all status report flags set and no custody transfer; D->C TCPCL; C->B UDPCL; B->A LTPCL. It is recommended that the initial test is with small bundles to eliminate fragmentation. A second iteration utilized larger bundles to ensure fragmentation.

In both test cases the test data generator recommended is DTNperf. The number of bundles is at least 500 for each iteration of tests.

Logs were be retrieved after each activity for analysis.

Expected Results

1. Users at Nodes A and D initiated tests. Test can be conducted simultaneously.
2. Status reports to the originating node were used to verify acceptance.
3. Bundles are received at each node and subsequently deleted from local store.
4. Bundles are forwarded.
5. If pathways are interrupted during test, bundle transfers were resume when restarted.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| TC0.a-1 | Start all 4 DTN nodes |  | Success |
| TC0.a-2 | Node D:  dtnsink –p –v -n 1 ipn:21000.2 | Prepare to receive 1 bundle | Success |
| TC0.a-3 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k | Send 1 bundle with 1000 byte payload. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (No bundle fragmentation) | Success |
| TC0.a-4 | Node B: Examine dtn.log | No indication of bundle fragmentation | Success |
| TC0.a-5 | Node D: Examine dtnsink output and dtn.log | 1 non-fragmented bundle should be received and status reports generated | Success |
| TC0.a-6 | Node A: Examine ion.log | Forwarding and delivery reports should have been logged | Success |
| TC0.a-7 | Node D:  dtnsink –p –v -n 600 ipn:21000.2 | Prepare to receive 600 bundles | Success |
| TC0.a-8 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 600k | Send 600 bundles with 1000 byte payload at 20 per second. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (No bundle fragmentation) | Success |
| TC0.a-9 | Node B: Examine dtn.log | No indication of bundle fragmentation | Success |
|  |  |  |  |
| TC0.a-10 | Node D: Examine dtnsink output and dtn.log | 600 non-fragmented bundles should be received | Success |
| TC0.a-11 | Node A: Examine ion.log | Forwarding and delivery reports should have been logged | Success |
| TC0.a-12 | Store log files, etc. |  | Success |
|  |  |  |  |
| TC0.a-13 | Stop and restart all 4 DTN nodes |  | Success |
| TC0.a-14 | Node D:  dtnsink –p –v -n 1 ipn:21000.2 | Prepare to receive 1 bundle | Success |
| TC0.a-15 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 10k –R 20b –D 10k | Send 1 bundle with 10,000 byte payload. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (Bundle fragmentation at Node B UDPCL) | Success |
| TC0.a-16 | Node B: Examine dtn.log  <Node C report fragments received?> | 8 bundle fragments should be generated and transmitted and status reports generated | Success |
| TC0.a-17 | Node D: Examine dtnsink output and dtn.log | 8 bundle fragments should be received and reconstituted into 1 bundle delivered to dtnsink and status reports generated | Success |
| TC0.a-18 | Node A: Examine ion.log | Forwarding and delivery reports should have been logged | Success |
| TC0.a-19 | Node D:  dtnsink –p –v -n 600 ipn:21000.2 | Prepare to receive 600 bundles | Success |
| TC0.a-20 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 10k –R 20b –D 6M | Send 600 bundles with 10,000 byte payload at 20 per second. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (Bundle fragmentation at Node B UDPCL) | Success |
| TC0.a-21 | Node B: Examine dtn.log  <Node C report fragments received?> | Bundle fragments should be generated and transmitted and status reports generated | Success |
| TC0.a-22 | Node D: Examine dtnsink output and dtn.log | 600 non-fragmented bundles should be received and status reports generated | Success |
| TC0.a-23 | Node A: Examine ion.log | Forwarding and delivery reports should have been logged | Success |
| TCO.a-24 | Store log files, etc. |  | Success |

**Table A-6: Basic Test Cases – Set A**

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| TC0.b-1 | Start all 4 DTN nodes |  | Success |
| TC0.b-2 | Node A:  bpsink ipn:17000.2 | Prepare to receive 1 bundle | Success |
| TC0.b-3 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k | Send 1 bundle with 1000 byte payload. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (No bundle fragmentation) | Success |
| TC0.b-4 | Node B: Examine dtn.log | No indication of bundle fragments received | Success |
| TC0.b-5 | Node A: Examine bpsink output and ion.log | 1 non-fragmented bundle should be received and status reports generated | Success |
| TC0.b-6 | Node D: Examine ion.log | Forwarding and delivery reports should have been logged | Success |
| TC0.b-7 | Node A:  bpsink ipn:17000.2 | Prepare to receive 600 bundles | Success |
| TC0.b-8 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 600k | Send 600 bundles with 1000 byte payload at 20 per second. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (No bundle fragmentation) | Success |
| TC0.b-9 | Node B: Examine dtn.log | No indication of bundle fragments received | Success |
| TC0.b-10 | Node A: Examine bpsink output and ion.log | 600 non-fragmented bundles should be received and status reports generated | Success |
| TC0.b-11 | Node D: Examine ion.log | Forwarding and delivery reports should have been logged | Success |
| TC0.b-12 | Store log files |  | Success |
|  |  |  |  |
| TC0.b-13 | Stop and restart all 4 DTN nodes |  | Success |
| TC0.b-14 | Node A:  bpsink ipn:17000.2 | Prepare to receive 1 bundle | Success |
| TC0.b-15 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 10k –R 20b –D 10k | Send 1 bundle with 10,000 byte payload. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (Bundle fragmentation at Node C UDPCL) | Success |
| TC0.b-16 | Node B: Examine dtn.log  <Node C report fragments generated?> | 8 bundle fragments should be received and status reports generated | Success |
|  |  |  |  |
| TC0.b-17 | Node A: Examine bpsink output and ion.log | 8 bundle fragments should be received and reconstituted into 1 bundle delivered to bpsink and status reports generated | Success |
| TC0.b-18 | Node D: Examine dtn.log | Forwarding and delivery reports should have been logged | Success |
| TC0.b-19 | Node A:  bpsink ipn:17000.2 | Prepare to receive 600 bundles | Success |
| TC0.b-20 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 10k –R 20b –D 6M | Send 600 bundles with 10,000 byte payload at 20 per second. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (Bundle fragmentation at Node C UDPCL) | Success |
| TC0.b-21 | Node B: Examine dtn.log  <Node C report fragments generated?> | Bundle fragments should be received and status reports generated | Success |
| TC0.b-22 | Node A: Examine bpsink output and ion.log | 600 reconstituted bundles should be received | Success |
| TC0.b-23 | Node D: Examine ion.log | Forwarding and delivery reports should have been logged | Success |
| TCO.b-24 | Store log files |  | Success |

**Table A-7: Basic Test Cases– Set B**

* + - 1. Basic Test Cases TC1.a and TC1.b

These test cases are characterized by 4 nodes in a linear topology (A, B, C, D). In the first test case (TC1.a), ION – A sendt valid bundles A->D with all status report flags set and custody transfer in the following order; A->B LTPCL; B->C UDPCL; C->D TCPCL. It is recommended that the initial test is with small bundles to eliminate fragmentation. A second iteration utilized larger bundles to ensure fragmentation.

In the second test case (TC1.b), DTN2 – D sent valid bundles D->A with all status report flags set and custody transfer; D->C TCPCL, C->B UDPCL; B->A LTPCL. It is recommended that the initial test is with small bundles to eliminate fragmentation. A second iteration utilized larger bundles to ensure fragmentation.

Aggregate Custody Signals were enabled between A and B.

In both test cases the test data generator recommended is DTNperf. The number of bundles is at least 500 for each iteration of tests.

Logs were be retrieved after each activity for analysis.

Expected Results

1. Users at Nodes A and D initiated tests. Test can be conducted simultaneously.
2. Status reports to the originating node were used to verify acceptance.
3. Bundles are received at each node and subsequently deleted from local store.
4. Bundles are forwarded.
5. If pathways are interrupted during test, bundles transfer were resumed when restarted.
6. ION-A receives/transmits aggregate custody signal (ACS) and bundles and verifies content.
7. ION-C receives/transmits custody signal and bundles.
8. DTN-B receives/transmits aggregate custody signal (ACS) and bundles.
9. DTN-D receives/transmits custody signal and bundles and verifies content.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| TC1.a-1 | Start all 4 DTN nodes |  | Success |
| TC1.a-2 | Enable ACS on Node A with Node B:  acsadmin  > 1 7 262144  > a ipn:19000.0 100 60 | Initialize ACS with Node B. Aggregate until size is 100 bytes or for 60 seconds. | Success |
| TC1.a-3 | Enable ACS on Node B:  > acs set enabled true  > acs set delay 60  > acs set size 100 | Initialize ACS with Node B. Aggregate until size is 100 bytes or for 60 seconds. | Success |
| TC1.a-4 | Node D:  dtnsink –p –v -n 1 ipn:21000.2 | Prepare to receive 1 bundle | Success |
| TC1.a-5 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k -C | Send 1 bundle with 1000 byte payload requesting custody transfer. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (No bundle fragmentation) | Success |
| TC1.a-6 | Wait 40 seconds | ACS time set to 60 seconds so no ACS should have been sent from B to A yet | Success |
| TC1.a-7 | Node A:  acslist | acslist output appended to ion.log should list 1 custody IDs | Success |
| TC1.a-8 | Node B:  > acs dump | ACS statistics should show 1 accepted and released (by regular CS from C) and 0 ACS generated | Success |
| TC1.a-9 | Wait 30 seconds | ACS should have been sent from B to A | Success |
| TC1.a-10 | Node A:  acslist | acslist output appended to ion.log should list 0 custody IDs | Success |
|  |  |  |  |
| TC1.a-11 | Node B:  > acs dump | ACS statistics should show 1 accepted and released (by regular CS from Node C) and 1 ACS generated | Success |
| TC1.a-12 | Node B: Examine dtn.log | No indication of bundle fragmentation | Success |
| TC1.a-13 | Node D: Examine dtnsink output and dtn.log | 1 non-fragmented bundle should be received and status reports generated | Success |
| TC1.a-14 | Node A: Examine ion.log | Forwarding, delivery and custody accepted reports should have been logged | Success |
| TC1.a-15 | Node D:  dtnsink –p -n 600 ipn:21000.2 | Prepare to receive 600 bundles | Success |
| TC1.a-16 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 600k –C | Send 600 bundles with 1000 byte payload requesting custody transfer at 20 per second. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (No bundle fragmentation) | Success |
| TC1.a-17 | Wait 40 seconds | ACS time set to 60 seconds so no ACS should have been sent from B to A yet | Success |
| TC1.a-18 | Node A:  acslist | acslist output appended to ion.log should list 600 custody IDs | Success |
| TC1.a-19 | Node B:  > acs dump | ACS statistics should show 601 accepted and released (by regular CS from Node C) and 1 previous ACS generated | Success |
| TC1.a-20 | Wait 30 seconds | ACS should have been sent from B to A | Success |
| TC1.a-21 | Node A:  acslist | acslist output appended to ion.log should list 0 custody IDs | Success |
| TC1.a-22 | Node B:  > acs dump | ACS statistics should show 601 accepted and released (by regular CS from Node C) and 2 ACS generated | Success |
| TC1.a-23 | Nodes B: Examine dtn.log | No indication of bundle fragmentation | Success |
| TC1.a-24 | Node D: Examine dtnsink output and dtn.log | 600 non-fragmented bundles should be received | Success |
| TC1.a-25 | Node A: Examine ion.log | Forwarding, delivery and custody accepted reports should have been logged | Success |
| TC1.a-26 | Save log files, etc. |  | Success |
|  |  |  |  |
| TC1.a-27 | Stop and restart all 4 DTN nodes |  | Success |
| TC1.a-28 | Enable ACS on Node A with Node B:  acsadmin  > 1 7 262144  > a ipn:19000.0 100 60 | Initialize ACS with Node B. Aggregate until size is 100 bytes or for 60 seconds. | Success |
| TC1.a-29 | Enable ACS on Node B:  > acs set enabled true  > acs set delay 60  > acs set size 100 | Initialize ACS with Node B. Aggregate until size is 100 bytes or for 60 seconds. | Success |
| TC1.a-30 | Node D:  dtnsink –p –v -n 1 ipn:21000.2 | Prepare to receive 1 bundle | Success |
| TC1.a-31 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 10k –R 20b –D 10k -C | Send 1 bundle with 10,000 byte payload requesting custody transfer. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (Bundle fragmentation at Node B UDPCL) | Success |
| TC1.a-32 | Wait 40 seconds | ACS time set to 60 seconds so no ACS should have been sent from B to A yet | Success |
| TC1.a-33 | Node A:  acslist | acslist output appended to ion.log should list 1 custody IDs | Success |
| TC1.a-34 | Node B:  > acs dump | ACS statistics should show 1 bundle accepted and released (by regular CS from C) and 0 ACS generated | Success |
| TC1.a-35 | Wait 30 seconds | ACS should have been sent from B to A | Success |
| TC1.a-36 | Node A:  acslist | acslist output appended to ion.log should list 0 custody IDs | Success |
| TC1.a-37 | Node B:  > acs dump | ACS statistics should show 1 accepted and released (by regular CS from Node C) and 1 ACS generated | Success |
| TC1.a-38 | Node B: Examine dtn.log  <Node C report fragments received?> | 8 bundle fragments should be generated and transmitted and status reports generated | Success |
| TC1.a-39 | Node D: Examine dtnsink output and dtn.log | 1 bundle should be received and status reports generated | Success |
| TC1.a-40 | Node A: Examine ion.log | Forwarding, delivery and custody accepted reports should have been logged | Success |
| TC1.a-41 | Node D:  dtnsink –p -n 600 ipn:21000.2 | Prepare to receive 600 bundles | Success |
|  |  |  |  |
| TC1.a-42 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 10k –R 20b –D 6M –C | Send 600 bundles with 10,000 byte payload requesting custody transfer at 20 per second. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (Bundle fragmentation at Node B UDPCL) | Success |
| TC1.a-43 | Wait 40 seconds | ACS time set to 60 seconds so no ACS should have been sent from B to A yet | Success |
| TC1.a-44 | Node A:  acslist | acslist output appended to ion.log should list 600 custody IDs | Success |
| TC1.a-45 | Node B:  > acs dump | ACS statistics should show 601 accepted and released (by regular CS from Node C) and 1 previous ACS generated | Success |
| TC1.a-46 | Wait 30 seconds | ACS should have been sent from B to A | Success |
| TC1.a-47 | Node A:  acslist | acslist output appended to ion.log should list 0 custody IDs | Success |
| TC1.a-48 | Node B:  > acs dump | ACS statistics should show 601 accepted and released (by regular CS from Node C) and 2 ACS generated | Success |
| TC1.a-49 | Node B: Examine dtn.log  <Node C report fragments received?> | Bundle fragments should be generated and transmitted and status reports generated | Success |
| TC1.a-50 | Node D: Examine dtnsink output and dtn.log | 600 non-fragmented bundles should be received | Success |
| TC1.a-51 | Node A: Examine ion.log | Forwarding, delivery and custody accepted reports should have been logged | Success |
| TC1.a-52 | Store log files, etc. |  | Success |

**Table A-8: Test Cases TC1.a**

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| TC1.b-1 | Start all 4 DTN nodes |  | Success |
| TC1.b-2 | Enable ACS on Node A with Node B:  acsadmin  > 1 7 262144  > a ipn:19000.0 100 60 | Initialize ACS with Node B. Aggregate until size is 100 bytes or for 60 seconds. | Success |
| TC1.b-3 | Enable ACS on Node B:  > acs set enabled true  > acs set delay 60  > acs set size 100 | Initialize ACS with Node A. Aggregate until size is 100 bytes or for 60 seconds. | Success |
| TC1.b-4 | Node A:  bpsink ipn:17000.2 | Prepare to receive 1 bundle | Success |
| TC1.b-5 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k –C | Send 1 bundle with 1000 byte payload requesting custody transfer. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (No bundle fragmentation) | Success |
| TC1.b-6 | Wait 40 seconds | ACS time set to 60 seconds so no ACS should have been sent from A to B yet | Success |
| TC1.b-7 | Node A:  acslist? | Is there a way to see aggregating custody IDs?? | Success |
| TC1.b-8 | Node B:  > acs dump | ACS statistics should show 1 bundle accepted and in custody | Success |
| TC1.b-9 | Wait 30 seconds | ACS should have been sent from A to B | Success |
| TC1.b-10 | Node A:  acslist? | Is there a way to see ACS was sent? | Success |
| TC1.b-11 | Node B:  > acs dump | ACS statistics should show 1 accepted and released by ACS | Success |
| TC1.b-12 | Node B: Examine dtn.log | No bundle fragmentation should be indicated | Success |
| TC1.b-13 | Node A: Examine bpsink output and ion.log | 1 unfragmented bundle should be received and status reports generated | Success |
| TC1.b-14 | Node D: Examine dtn.log | Forwarding, delivery and custody accepted reports should have been logged | Success |
| TC1.b-15 | Node A:  bpsink ipn:17000.2 | Prepare to receive 600 bundles | Success |
| TC1.b-16 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 600k –C | Send 600 bundles with 1000 byte payload requesting custody transfer at 20 per second. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (No bundle fragmentation) | Success |
| TC1.b-17 | Wait 40 seconds | ACS time set to 60 seconds so no ACS should have been sent from A to B yet | Success |
| TC1.b-18 | Node A:  acslist? | Is there a way to see aggregating custody IDs?? | Success |
| TC1.b-19 | Node B:  > acs dump | ACS statistics should show 600 bundles in custody and 1 previously released by ACS | Success |
| TC1.b-20 | Wait 30 seconds | ACS should have been sent from A to B | Success |
| TC1.b-21 | Node A:  acslist? | Is there a way to see ACS was sent? | Success |
| TC1.b-22 | Node B:  > acs dump | ACS statistics should show 601 accepted and released by ACS | Success |
| TC1.b-23 | Node B: Examine dtn.log | No indication of bundle fragmentation | Success |
| TC1.b-24 | Node A: Examine bpsink output and ion.log | 600 non-fragmented bundles should be received | Success |
| TC1.b-25 | Node D: Examine dtn.log | Forwarding, delivery and custody accepted reports should have been logged | Success |
| TC1.b-26 | Save log files, etc. |  | Success |
|  |  |  |  |
| TC1.b-27 | Stop and restart all 4 DTN nodes |  | Success |
| TC1.b-28 | Enable ACS on Node A with Node B:  acsadmin  > 1 7 262144  > a ipn:19000.0 100 60 | Initialize ACS with Node B. Aggregate until size is 100 bytes or for 60 seconds. | Success |
| TC1.b-29 | Enable ACS on Node B:  > acs set enabled true  > acs set delay 60  > acs set size 100 | Initialize ACS with Node A. Aggregate until size is 100 bytes or for 60 seconds. | Success |
| TC1.b-30 | Node A:  bpsink ipn:17000.2 | Prepare to receive 1 bundle | Success |
| TC1.b-31 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 10k –R 20b –D 10k -C | Send 1 bundle with 10,000 byte payload requesting custody transfer. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (Bundle fragmentation at Node C UDPCL) | Success |
| TC1.b-32 | Wait 40 seconds | ACS time set to 60 seconds so no ACS should have been sent from B to A yet | Success |
| TC1.b-33 | Node A:  acslist? | Is there a way to see aggregating custody IDs?? | Success |
| TC1.b-34 | Node B:  > acs dump | ACS statistics should show 1 bundle accepted and in custody (or 8 for each fragment?) | Success |
| TC1.b-35 | Wait 30 seconds | ACS should have been sent from B to A | Success |
| TC1.b-36 | Node A:  acslist? | Is there a way to see ACS was sent? | Success |
| TC1.b-37 | Node B:  > acs dump | ACS statistics should show 1 accepted and released by ACS | Success |
| TC1.b-38 | Node B: Examine dtn.log | 8 bundle fragments should have been received | Success |
| TC1.b-39 | Node A: Examine bpsink output and ion.log | 1 bundle should be received and status reports generated | Success |
| TC1.b-40 | Node D: Examine dtn.log | Forwarding, delivery and custody accepted reports should have been logged | Success |
| TC1.b-41 | Node A:  bpsink ipn:17000.2 | Prepare to receive 600 bundles | Success |
| TC1.b-42 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 10k –R 20b –D 6M –C | Send 600 bundles with 10,000 byte payload requesting custody transfer at 20 per second. Status reports will be sent to ipn:21000.0 and logged to ion.log.  (Bundle fragmentation at Node C UDPCL) | Success |
| TC1.b-43 | Wait 40 seconds | ACS time set to 60 seconds so no ACS should have been sent from A to B yet | Success |
| TC1.b-44 | Node A:  acslist? | Is there a way to see aggregating custody IDs?? | Success |
| TC1.b-45 | Node B:  > acs dump | ACS statistics should show 600 bundles in custody and 1 previously released by ACS | Success |
| TC1.b-46 | Wait 30 seconds | ACS should have been sent from B to A | Success |
| TC1.b-47 | Node A:  acslist? | Is there a way to see ACS was sent? | Success |
| TC1.b-48 | Node B:  > acs dump | ACS statistics should show 601 accepted and released by ACS | Success |
| TC1.b-49 | Nodes B: Examine dtn.log | Bundle fragments should have been received and status reports generated | Success |
| TC1.b-50 | Node A: Examine bpsink output and ion.log | 600 bundles should be received | Success |
| TC1.b-51 | Node D: Examine dtn.log | Forwarding, delivery and custody accepted reports should have been logged | Success |
| TC1.b-52 | Store log files, etc. |  | Success |

**Table A-8: Test Cases TC1.b**

* + 1. Failure Test Cases Procedures

The following test cases exercise various scenarios which demonstrate the behavior of a DTN router. The router behavior is not characteristic of ION or DTN2 implementations. However to exercise the capabilities participation by ION and DTN2 nodes is required. There are four basic failure cases that are repeated depending whether ION or DTN2 are initiating or terminating nodes.

* + - 1. Failure Test Cases CF0.a and CF0.b

The purpose of these test cases is to exercises the behavior of three DTN nodes when one node has no route to a third non-existent node. The two test cases are equivalent in their methodology but reverse roles between DTN2 and ION. Test case CF0.a originates a bundle with an ION node and a DTN2 node must disposition the bundle. Test case CF0.b has a DTN2 node originating a bundle and an ION node dispositioning the bundle. Tests were conducted with and without custody transfer requested. Logs were be retrieved after each activity for analysis.

The data flow is depicted in Figure 6-2 and the node configuration is detailed in Table 6-2.

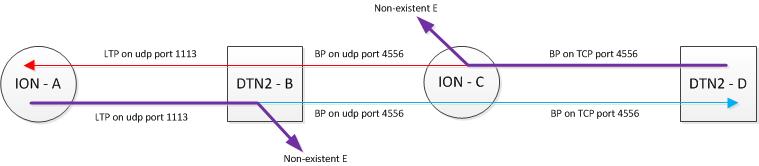


Figure A-5: Failure Test Cases CF0.a and CF0.b – Data Flow

| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| --- | --- | --- | --- | --- | --- |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0  3333 3333 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0  3333 3333 ipn:20000.0 |

Table A-9 Failure Test Cases CF0.a and CF0.b – Node Configuration

Expected Results

1. Users at Nodes A and D initiated tests. Tests can be conducted simultaneously.
2. The global Bundle ID was used to verify bundle identities.
3. Bundles will expire while being held and awaiting a route to a non-existent node E.
4. Custody acceptance may be refused per an implementation decision

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| CF0.a-1 | Start all 4 DTN nodes |  | Success |
| CF0.a-2 | Node A:  dtnperf\_vION --client –r –f --del  –l 60 -m ipn:17000.0  -d ipn:3333.2  –P 1k –R 20b –D 1k | Send 1 bundle with 1000 byte payload and lifetime 60 seconds to routed but non-existent node 3333. Status reports will be sent to ipn:17000.0 and logged to ion.log. | Success |
| CF0.a-3 | Wait 30 seconds | Bundle should not have expired yet | Success |
| CF0.a-4 | Node B:  > bundle list | 1 bundle should be pending and can be examined | Success |
| CF0.a-5 | Wait 40 seconds | Bundle should have expired | Success |
| CF0.a-6 | Node B:  > bundle list | 1 bundle should have expired | Success |
| CF0.a-7 | Node A: Examine ion.log | Status reports should have been logged | Success |
| CF0.a-8 | Node A:  dtnperf\_vION --client –r –f --del  –l 60 -m ipn:17000.0  -d ipn:3333.2  –P 1k –R 20b –D 1k -C | Send 1 bundle with 1000 byte payload with custody transfer requested and lifetime 60 seconds to routed but non-existent node 3333. Status reports will be sent to ipn:17000.0 and logged to ion.log. | Success |
| CF0.a-9 | Wait 30 seconds | Bundle should not have expired yet | Success |
| CF0.a-10 | Node B:  > bundle list | 1 bundle should be pending and in custody and can be examined | Success |
| CF0.a-11 | Wait 40 seconds | Bundle should have expired | Success |
| CF0.a-12 | Node B:  > bundle list | 1 bundle should have expired | Success |
| CF0.a-13 | Node A: Examine ion.log | Status reports should have been logged | Success |
| CF0.a-14 | Save log files, etc. |  | Success |

**Table A-10: Test Cases CFO.a**

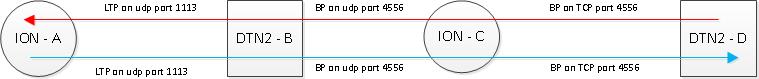
| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| CF0.b-1 | Start all 4 DTN nodes |  | Success |
| CF0.b-2 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 60 -m ipn:21000.0  -d ipn:3333.2  –P 1k –R 20b –D 1k | Send 1 bundle with 1000 byte payload and lifetime 60 seconds to routed but non-existent node 3333. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| CF0.b-3 | Wait 30 seconds | Bundle should not have expired yet | Success |
| CF0.b-4 | Node C:  bplist | 1 bundle may be listed or may have been deleted? | Success |
| CF0.b-5 | Wait 40 seconds | Bundle should have expired | Success |
| CF0.b-6 | Node C:  bplist | No bundles listed | Success |
| CF0.b-7 | Node D: Examine dtn.log | Status reports should have been logged | Success |
| CF0.b-8 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 60 -m ipn:21000.0  -d ipn:3333.2  –P 1k –R 20b –D 1k -C | Send 1 bundle with 1000 byte payload with custody transfer requested and lifetime 60 seconds to routed but non-existent node 3333. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| CF0.b-9 | Wait 30 seconds | Bundle should not have expired yet | Success |
| CF0.b-10 | Node C:  bplist | 1 bundle may be listed and may be in custody or may have been deleted? | Success |
| CF0.b-11 | Wait 40 seconds | Bundle should have expired | Success |
| CF0.b-12 | Node C:  bplist | No bundles listed | Success |
| CF0.b-13 | Node D: Examine dtn.log | Status reports should have been logged | Success |
| CF0.b-14 | Save log files, etc. |  | Success |

**Table A-11: Test Cases CFO.b**

* + - 1. Failure Test Cases CF0.c and CF0.d

The purpose of these test cases is to exercise and verify the behavior of DTN nodes when the source endpoint ID of “dtn:none”. The two test cases are equivalent in their methodology but reverse roles between DTN2 and ION. Test case CF0.c originates a bundle with an ION node whereas test case CF0.d has a DTN2 node originating a bundle. Logs were be retrieved after each activity for analysis.

The data flow is depicted in Figure 6-3 and the node configuration is detailed in Table 6-3.



**Figure A-6: Failure Test Cases CF0.c and CF0.d – Data Flow**

| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| --- | --- | --- | --- | --- | --- |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0 |

**Table A-12: Failure Test Cases CF0.c and CF0.d – Node Configuration**

Expected Results

1. Users at Nodes A and D initiated tests. Test can be conducted simultaneously.
2. Router/Bundle behavior must be consistent with RFC 5050 section 4.2.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| CF0.c-1 | Start all 4 DTN nodes |  | Success |
| CF0.c-2 | Node D:  dtnsink –p –v -n 1 ipn:21000.2 | Prepare to receive 1 bundle | Success |
| CF0.c-3 | Node A:  bpsource ipn:21000.2 “test” | Send 1 bundle with a NULL source EID  (currently no way to request status reports or custody transfer) | Success |
| CF0.c-4 | Node D: Examine dtnsink output and dtn.log | 1 bundle should have been received | Success |
| CF0.c-5 | Node A: Examine ion.log | Status reports should have been logged | Success |
| CF0.c-6 | Save log files, etc. |  | Success |

**Table A-12: Test Cases CFO.c**

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| CF0.d-1 | Start all 4 DTN nodes |  | Success |
| CF0.d-2 | Node A:  bpsink ipn:17000.2 | Prepare to receive 1 bundle | Success |
| CF0.d-3 | Node D:  dtnsend –s dtn:none –d ipn:17000.2  –t m –p test | Send 1 bundle with a NULL source EID allowing fragmentation | Success |
| CF0.d-4 | Node D: Examine dtn.log | Bundle should be rejected due to allowing fragmentation | Success |
| CF0.d-5 | Node D:  dtnsend –s dtn:none –d ipn:17000.2  –t m –p test –W -c | Send 1 bundle with a NULL source EID with fragmentation disabled and requesting custody transfer | Success |
| CF0.d-6 | Node D: Examine dtn.log | Bundle should be rejected due to requesting custody transfer | Success |
| CF0.d-7 | Node D:  dtnsend –s dtn:none –d ipn:17000.2  –t m –p test –W -R | Send 1 bundle with a NULL source EID with fragmentation disabled and requesting reception receipts | Success |
| CF0.d-8 | Node D: Examine dtn.log | Bundle should be rejected due to requesting status report | Success |
| CF0.d-9 | Node D:  dtnsend –s dtn:none –d ipn:17000.2  –t m –p test –W | Send 1 bundle with a NULL source EID with fragmentation disabled | Success |
| CF0.d-10 | Node A: Examine bpsink output | 1 bundle should be received | Success |
| CF0.c-11 | Save log files, etc. |  | Success |

**Table A-13: Test Cases CFO.d**

* + - 1. Failure Test Cases CF0.e and CF0.f

The purpose of these test cases is to exercise and verify the behavior of DTN bundle expiration. These two test cases are equivalent in their methodology but reverse roles between DTN2 and ION. Test case CF0.e originates a bundle with an ION node whereas test case CF0.f has a DTN2 node originating a bundle. Logs were be retrieved after each activity for analysis.

The data flow is depicted in Figure 6-4 and the node configuration is detailed in Table 6-4.

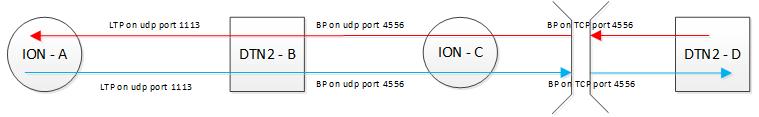


Figure A-7: Failure Test Cases CF0.e and CF0.f – Data Flow

| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| --- | --- | --- | --- | --- | --- |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0 |

Table A-14: Failure Test Cases CF0.e and CF0.f – Node Configuration

Expected Results

1. Users at Nodes A and D will initiate tests. Test cannot be conducted simultaneously.
2. Router/Bundle behavior must be consistent with RFC 5050 section 5.5.
3. Bundles will expire while being held and awaiting a route to an adjacent node.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| CF0.e-1 | Start DTN nodes A, B and C |  | Success |
| CF0.e-2 | Node A:  dtnperf\_vION --client –r –f --del  –l 60 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k | Send 1 bundle with 1000 byte payload and lifetime 60 seconds to Node D which is offline. Status reports will be sent to ipn:17000.0 and logged to ion.log. | Success |
| CF0.e-3 | Wait 30 seconds | Bundle should not have expired yet | Success |
| CF0.e-4 | Node C:  bplist | 1 bundle should be listed | Success |
| CF0.e-5 | Wait 40 seconds | Bundle should have expired | Success |
| CF0.e-6 | Node C:  bplist | No bundles should be listed | Success |
| CF0.e-7 | Node A: Examine ion.log | Status reports should have been logged | Success |
| CF0.e-8 | Save log files, etc. |  | Success |

**Table A-15: Test Cases CFO.e**

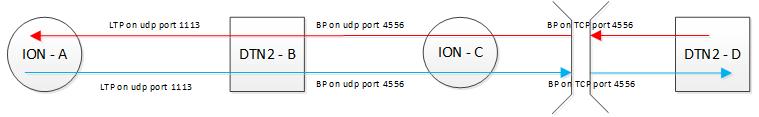
| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| CF0.f-1 | Start only DTN node D |  | Success |
| CF0.f-2 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 60 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k | Send 1 bundle with 1000 byte payload and lifetime 60 seconds to Node A while Node C is offline. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| CF0.f-3 | Wait 30 seconds | Bundle should not have expired yet | Success |
| CF0.f-4 | Node D:  > bundle list | 1 bundle should be pending and can be examined | Success |
| CF0.f-5 | Wait 40 seconds | Bundle should have expired | Success |
| CF0.f-6 | Node D:  > bundle list | No bundles listed | Success |
| CF0.f-7 | Node D: Examine dtn.log | Status reports should have been logged | Success |
| CF0.f-8 | Save log files, etc. |  | Success |

Table A-16: Test Cases CFO.f

* + - 1. Failure Test Cases CF0.g and CF0.h

The purpose of these test cases is to exercise and verify the behavior of DTN bundle cancellation. These two test cases are equivalent in their methodology but reverse roles between DTN2 and ION. Test case CF0.g originates a bundle with an ION node whereas test case CF0.h has a DTN2 node originating a bundle. Logs were be retrieved after each activity for analysis.

The data flow is depicted in Figure 6-5 and the node configuration is detailed in Table 6-5.



**Figure A-8: Failure Test Cases CF0.g and CF0.h – Data Flow**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0 |

**Table A-17: Failure Test Cases CF0.g and CF0.h – Node Configuration**

Expected Results

1. Users at Nodes C and D will initiate tests. Test cannot be conducted simultaneously.
2. Router/Bundle behavior must be consistent with RFC 5050 section 5.13 and 5.15.
3. Bundles will be canceled while being held and awaiting a route to the adjacent node.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| CF0.g-1 | Start only DTN node C |  | Success |
| CF0.g-2 | Node C:  bptrace ipn:20000.2 ipn:21000.1 ipn:20000.0 3600 0.1 "test" rcv,fwd,dlv,del | Send 1 bundle with lifetime 3600 seconds to Node D which is offline. Status reports will be sent to ipn:20000.0 and logged to ion.log. | Success |
| CF0.g-3 | Wait 30 seconds | Bundle should not have expired yet | Success |
| CF0.g-4 | Node C:  bplist | 1 bundle should be listed | Success |
| CF0.g-5 | Node C:  bpcancel ipn:20000.2 <creation time> | Manually cancel the bundle with info from the bplist | Success |
| CF0.g-6 | Node C:  bplist | No bundles should be listed | Success |
| CF0.g-7 | Node C: Examine ion.log | Status reports should have been logged | Success |
| CF0.g-8 | Save log files, etc. |  | Success |

Table A-18: Test Cases CFO.g

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| CF0.h-1 | Configure Node B LTP for 1 session and transmission rate for 8 Kbps | Slow transmission rate so that bundles will queue up on the LTP CLA | Success |
| CF0.h-2 | Start all 4 DTN nodes |  | Success |
| CF0.h-2 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 20k | Send 20 bundles with 1000 byte payload and lifetime 3600 seconds to Node A. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| CF0.h-3 | Wait 3 seconds | Bundle should have queued up on Node B | Success |
| CF0.h-4 | Node B:  > bundle list | Several bundles should be pending and can be examined | Success |
| CF0.h-5 | Node B:  > bundle cancel <id> <link> | Manually cancel bundle(s) with info from the bundle list | Success |
| CF0.h-6 | Node B:  > bundle list | Canceled bundle(s) should be deleted | Success |
| CF0.h-7 | Node D: Examine dtn.log | Status reports should have been logged with reason “transmission canceled” | Success |
| CF0.h-8 | Save log files, etc. |  | Success |

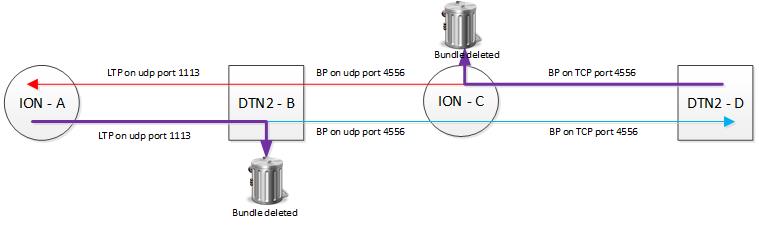
Table A-19: Test Cases CFO.h

* + 1. Bad Data Cases Procedures

The following test cases exercise various scenarios which demonstrate the behavior of a DTN router when an unintelligible block is received. There are two bad data cases to test ION and DTN2 behavior.

* + - 1. Bad Data Cases BD0.a and BD0.b

The purpose of these test cases is to exercise the behavior of a DTN node when a bundle is received that contains a block that it cannot process and which has the delete bundle on error flag set. The two test cases are equivalent in their methodology but reverse roles between DTN2 and ION. Test case CF0.a originates a bundle with an ION node and a DTN2 node must disposition the bundle. Test case CF0.b has a DTN2 node originating a bundle and an ION node dispositioning the bundle. Logs were retrieved after each activity for analysis.

The data flow is depicted in Figure 6-6 and the node configuration is detailed in Table 6-6.

**Figure A-9: Bad Data Test Cases BD0.a and BD0.b – Data Flow**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0 |

**Table A-20: Bad Data Test Cases BD0.a and BD0.b – Node Configuration**

Expected Results

1. Users at Nodes A and D will initiate tests. Test cannot be conducted simultaneously.
2. Router/Bundle behavior must be consistent with RFC 5050 section 5.6 and 5.13.
3. Bundles will be deleted for the reason “Block unintelligible”.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| BD0.a-1 | Start all 4 DTN nodes |  | Success |
| BD0.a-2 | Node D:  dtnsink –p -v -n 1 ipn:21000.2 | Prepare to receive 1 bundle | Success |
| BD0.a-3 | Node A:  export CCSDS\_TEST=1 | Set environment variable to set delete on error bit in the Bundle Age Extension block  (special code for this test) | Success |
| BD0.a-4 | Node A:  dtnperf\_vION --client –r –f --del  –l 60 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k | Send 1 bundle with 1000 byte payload and delete on error bit set in age block. Status reports will be sent to ipn:17000.0 and logged to ion.log. | Success |
| BD0.a-5 | Node B: Examine dtn.log | Bundle should be deleted due to unknown block type with the delete on error bit set | Success |
| BD0.a-6 | Node D: Examine dtnsink output | No bundle should be received | Success |
| BD0.a-7 | Node A: Examine ion.log | Status reports should have been logged | Success |
| BD0.a-8 | Save log files, etc. |  | Success |
| BD0.a-9 | Node A:  export –n CCSDS\_TEST  unset CCSDS\_TEST | Remove environment variable | Success |

Table A-21: Test Cases BD0.a

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| BD0.b-1 | Start all 4 DTN nodes |  | Success |
| BD0.b-2 | Node A:  bpsink ipn:17000.2 | Prepare to receive 1 bundle | Success |
| BD0.b-3 | Node D:  > block set age\_outbound\_enabled true | Set DTN2 variable to set delete on error bit in a block that ION will not recognize  (special code for this test) | Success |
| BD0.b-4 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 60 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k | Send 1 bundle with 1000 byte payload and delete on error bit set in age block. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| BD0.b-5 | Node C: Examine ion.log | Bundle should be deleted due to unknown block type with the delete on error bit set | Success |
| BD0.b-6 | Node A: Examine bpsink output | No bundle should be received | Success |
| BD0.b-7 | Node D: Examine dtn.log | Status reports should have been logged | Success |
| BD0.b-8 | Save log files, etc. |  | Success |
| BD0.b-9 | Node D:  > block set age\_outbound\_enabled false | Remove DTN2 variable | Success |

Table A-22: Test Cases BD0.b

* + 1. Other Test Cases Procedures

Ten additional test cases are proposed to test capabilities and are shown in Table 5-3. They include unique LTP and ECOS features. The tests required unique configurations. Two locales were necessary. Test cases OTH.a through OTH.h were conducted across the Internet as previously denoted in section 2. However due to the unique characteristics of test cases OTH.i and OTH.j and the possibility of VPN congestion, these test cases were conducted at the CCSDS meeting at CalTech in Pasadena, California, USA on March 25th.

* + - 1. LTP SDA aggregation timeout cases OTH.a and OTH.b

The purpose of these test cases is to exercises the behavior of DTN nodes with bundles encapsulated in LTP. An insufficient number of bundles were transmitted to initiate service data aggregation (SDA) of an LTP segment. As a result, the segment will experience timeout and the LTP segment was forwarded to the next node. The two test cases are equivalent in their methodology but reverse roles between DTN2 and ION. Test case OTH.a originates bundles in LTP segments with an ION node and a DTN2 node must disposition the segment and bundles. Test case OTH.b has a DTN2 node originating bundles in LTP segments and an ION node dispositioning the segment and bundles. Tests were conducted without custody transfer. Logs were be retrieved after each activity for analysis.

The data flow is depicted in Figure 6-7 and the node configuration is detailed in Table 6-7.

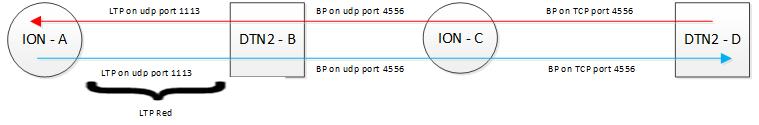


Figure A-10: SDA Aggregation Test Case OTH.a and OTH.b - Data Flow

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0 |

Table A-23: SDA Aggregation Test Case OTH.a and OTH.b – Node Configuration

Additional Special Configuration

Node A and Node B shall configure the LTP CL aggregation time to 30 seconds and the aggregation size to 20,000 bytes.

Expected Results

1. Users at Nodes A and D will initiate tests. Test can be conducted simultaneously.
2. Only red LTP segments will be transmitted.
3. LTP segments will not be forwarded to adjacent routers until SDA timeout .
4. Bundles will be forwarded when de-encapsulated.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.a-1 | Start all 4 DTN nodes |  | Success |
| OTH.a-2 | Node D:  dtnsink –p –v -n 35 ipn:21000.2 | Prepare to receive 35 bundles | Success |
| OTH.a-3 | Node A:  dtnperf\_vION --client –r –f --del  –l 300 -m ipn:17000.0  -d ipn:21000.2  –P 100B –R 1b –D 3500B | Send 35 bundles with 100 byte payload at a rate of 1 bundle per second. Status reports will be sent to ipn:17000.0 and logged to ion.log. | Success |
| OTH.a-4 | Node D: Monitor dtnsink output for 30 seconds | No bundles should be received during the SDA aggregation time | Success |
| OTH.a-5 | Node D: Monitor dtnsink output a bit after the 30 second mark | 29 or 30 bundles should be received in a burst | Success |
| OTH.a-6 | Node D: Monitor dtnsink output for another 30 seconds | No bundles should be received during the SDA aggregation time | Success |
| OTH.a-7 | Node D: Monitor dtnsink output a bit after the 60 second mark | 5 or 6 bundles should be received in a burst | Success |
| OTH.a-8 | Wait 40 seconds | Allow time for status reports to be returned to Node A | Success |
| OTH.a-9 | Node A: Examine ion.log | Status reports should have been logged | Success |
| OTH.a-10 | Save log files, etc. |  | Success |

Table A-24: Test Cases OTH.a

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.b-1 | Start all 4 DTN nodes |  | Success |
| OTH.b-2 | Node A:  bpsink ipn:17000.2 | Prepare to receive 35 bundles | Success |
| OTH.b-3 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 300 -m ipn:21000.0  -d ipn:17000.2  –P 100B –R 1b –D 3500B | Send 35 bundles with 100 byte payload at a rate of 1 bundle per second. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| OTH.b-4 | Node A: Monitor bpsink output for 30 seconds | No bundles should be received during the SDA aggregation time | Success |
| OTH.b-5 | Node A: Monitor bpsink output a bit after the 30 second mark | 29 or 30 bundles should be received in a burst | Success |
| OTH.b-6 | Node A: Monitor bpsink output for another 30 seconds | No bundles should be received during the SDA aggregation time | Success |
| OTH.b-7 | Node A: Monitor bpsink output a bit after the 60 second mark | 5 or 6 bundles should be received in a burst | Success |
| OTH.b-8 | Wait 40 seconds | Allow time for status reports to be returned to Node A | Success |
| OTH.b-9 | Node D: Examine dtn.log | Status reports should have been logged | Success |
| OTH.b-10 | Save log files, etc. |  | Success |

Table A-25: Test Cases OTH.b

* + - 1. LTP SDA Aggregation induced transmission cases OTH.c and OTH.d

The purpose of these test cases is to re-execute test cases OTH.a and OTH.b steps with larger bundles to trigger SDA size aggregation of an LTP segment prior to the aggregation time. As a result, ION and DTN2 routers spontaneously transmit LTP segments to the next node. The two test cases are equivalent in their methodology but reverse roles between DTN2 and ION. Test case OTH.c originates bundles in LTP segments with an ION node and a DTN2 node must disposition the segment and bundles. Test case OTH.d has a DTN2 node originating bundles in LTP segments and an ION node dispositioning the segment and bundles. Tests were conducted without custody transfer. Logs were retrieved after each activity for analysis.

The data flow is depicted in Figure 6-8 and the node configuration is detailed in Table 6-8.

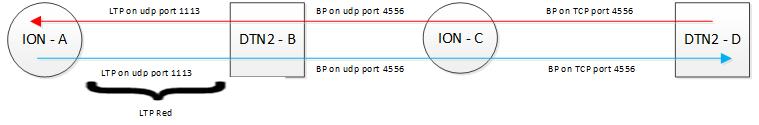


Figure A-11: SDA Aggregation Test Case OTH.c and OTH.d - Data Flow

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0 |

Table A-26: SDA Aggregation Test Case OTH.c and OTH.d – Node Configuration

Additional Special Configuration

Node A and Node B shall configure the LTP CL aggregation time to 30 seconds and the aggregation size to 20,000 bytes.

Expected Results

1. Users at Nodes A and D will initiate tests. Test can be conducted simultaneously.
2. Only red LTP segments will be transmitted.
3. LTP segments will be forwarded to adjacent routers prior to the SDA timeout .
4. Bundles will be forwarded when de-encapsulated.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.c-1 | Start all 4 DTN nodes |  | Success |
| OTH.c-2 | Node D:  dtnsink –p –v -n 35 ipn:21000.2 | Prepare to receive 35 bundles | Success |
| OTH.c-3 | Node A:  dtnperf\_vION --client –r –f --del  –l 300 -m ipn:17000.0  -d ipn:21000.2  –P 10k –R 1b –D 350k | Send 35 bundles with 10,000 byte payload at a rate of 1 bundle per second. Status reports will be sent to ipn:17000.0 and logged to ion.log. | Success |
| OTH.c-4 | Node D: Monitor dtnsink output for 34 seconds | Two bundles should be received in a burst approximately every 2 seconds | Success |
| OTH.c-5 | Node D: Monitor dtnsink output for an additional 30+ seconds | The final bundle should be received after approximately 30 seconds | Success |
| OTH.c-6 | Wait 40 seconds | Allow time for status reports to be returned to Node A | Success |
| OTH.c-7 | Node A: Examine ion.log | Status reports should have been logged | Success |
| OTH.c-8 | Save log files, etc. |  | Success |

Table A-27: Test Cases OTH.c

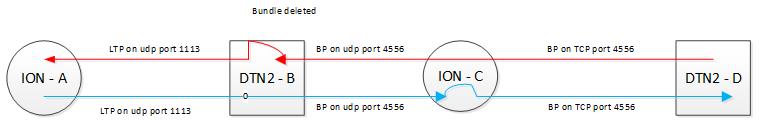
| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.d-1 | Start all 4 DTN nodes |  | Success |
| OTH.d-2 | Node A:  bpsink ipn:17000.2 | Prepare to receive 35 bundles | Success |
| OTH.d-3 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 300 -m ipn:21000.0  -d ipn:17000.2  –P 10k –R 1b –D 350k | Send 35 bundles with 10,000 byte payload at a rate of 1 bundle per second. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| OTH.d-4 | Node A: Monitor bpsink output for 34 seconds | Two bundles should be received in a burst approximately every 2 seconds | Success |
| OTH.d-5 | Node A: Monitor bpsink output for an additional 30+ seconds | The final bundle should be received after approximately 30 seconds | Success |
| OTH.d-6 | Wait 40 seconds | Allow time for status reports to be returned to Node A | Success |
| OTH.d-7 | Node A: Examine ion.log | Status reports should have been logged | Success |
| OTH.d-8 | Save log files, etc. |  | Success |

Table A-28: Test Cases OTH.d

* + - 1. ECOS bundle reordering cases OTH.e and OTH.f

The purpose of these test cases is to execute and verify the behavior of bundle reordering with DTN nodes when Extended Class of Service (ECOS) is in effect. Service was interrupted between intervening routers at which time a number of bundles of different classes were transmitted between ION and DTN2 routers. Bundle classes were sent in a chaotic order; neither ascending nor descending in class. When service is restored between adjacent routers, class of bundles sent were monitored for the correct order. The two cases are equivalent in their methodology but reverse roles between DTN2 and ION. Tests were conducted without custody transfer. Logs were retrieved after each activity for analysis.

The data flow is depicted in Figure 6-9 and the node configuration is detailed in Table 6-9.

 Figure A-12: ECOS bundle reordering cases OTH.e and OTH.f – Data Flow

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0 |

Table A-29: ECOS bundle reordering cases OTH.e and OTH.f– Node Configuration

Expected Results

1. Users at Nodes A and D will initiate tests. Test cannot be conducted simultaneously.
2. A minimum of three bundles is required,
3. Bundles will be forwarded in the correct class sequence.

Additional Special Configuration

Node B will utilize a NASA developed DTN2 External Router during this test as the built in static router does not currently support ECOS priority routing.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.e-1 | Start DTN nodes A, B and C |  | Success |
| OTH.e-2 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k  -p expedited –ordinal 101 | Send 1 bundle with expedited ECOS priority 101 and lifetime 3600 seconds to Node D which is offline. Status reports will be sent to ipn:17000.0 and logged to ion.log. | Success |
| OTH.e-3 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k  -p bulk –ordinal 200 | Send 1 bundle with bulk priority and ECOS ordinal 200  (ECOS ordinal should be ignored as it only applies to expedited) | Success |
| OTH.e-4 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k  -p normal –ordinal 153 | Send 1 bundle with normal priority and ECOS ordinal 153  (ECOS ordinal should be ignored as it only applies to expedited) | Success |
| OTH.e-5 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k  -p expedited –ordinal 42 | Send 1 bundle with expedited priority and ECOS ordinal 42 | Success |
| OTH.e-6 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k  -p expedited –ordinal 254 | Send 1 bundle with expedited priority and ECOS ordinal 254 | Success |
| OTH.e-7 | Node A:  dtnperf\_vION --client –r –f --del  –l 3600 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 20b –D 1k  -p expedited –ordinal 178 | Send 1 bundle with expedited priority and ECOS ordinal 178 | Success |
| OTH.e-8 | Start wireshark or tcpdump capture between Nodes C and D |  | Success |
| OTH.e-9 | Start DTN node D |  | Success |
| OTH.e-10 | Wait for Node C to connect to Node D | This can take a minute or more depending on timing | Success |
| OTH.e-11 | Node D:  > bundle list | Verify 6 bundles received and examine details to determine the order they were received | Success |
| OTH.e-12 | Stop network capture(s) and examine | Bundles should have been transmitted in order:  ECOS ordinal 254  ECOS ordinal 178  ECOS ordinal 101  ECOS ordinal 42  Normal  Bulk | Success |
| OTH.e-13 | Node A: Examine ion.log | Status reports should have been logged | Success |
| OTH.e-14 | Save log files, etc. |  | Success |

Table A-29: Test Cases OTH.e

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.f-1 | Start all 4 DTN nodes |  | Success |
| OTH.f-2 | Node A:  bpsink ipn:17000.2 | Prepare to receive bundles | Success |
| OTH.f-3 | Node B:  > configure LTP CLA as “LOS” | Simulate Node A is out of contact range so no bundles will transmit | Success |
| OTH.f-4 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k  -p expedited –ordinal 101 | Send 1 bundle with expedited ECOS priority 101 and lifetime 3600 seconds to Node D which is offline. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| OTH.f-5 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k  -p bulk –ordinal 200 | Send 1 bundle with bulk priority and ECOS ordinal 200  (ECOS ordinal should be ignored as it only applies to expedited) | Success |
| OTH.f-6 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k  -p normal –ordinal 153 | Send 1 bundle with normal priority and ECOS ordinal 153  (ECOS ordinal should be ignored as it only applies to expedited) | Success |
| OTH.f-7 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k  -p expedited –ordinal 42 | Send 1 bundle with expedited priority and ECOS ordinal 42 | Success |
| OTH.f-8 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k  -p expedited –ordinal 254 | Send 1 bundle with expedited priority and ECOS ordinal 254 | Success |
| OTH.f-9 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k  -p expedited –ordinal 178 | Send 1 bundle with expedited priority and ECOS ordinal 178 | Success |
| OTH.f-10 | Start wireshark or tcpdump capture between Nodes A and B |  | Success |
| OTH.f-11 | Node B:  > configure LTP CLA as “AOS” | Blocked bundles should be transmitted | Success |
| OTH.f-12 | Node D:  > bundle list | Verify 6 bundles received and examine details to determine the order they were received | Success |
| OTH.f-13 | Stop network capture(s) and examine | Bundles should have been transmitted in order:  ECOS ordinal 254  ECOS ordinal 178  ECOS ordinal 101  ECOS ordinal 42  Normal  Bulk | Success |
| OTH.f-14 | Node A: Examine ion.log | Status reports should have been logged | Success |
| OTH.f-15 | Save log files, etc. |  | Success |

**Table A-30: Test Cases OTH.f**

* + - 1. LTP Green Transmission

The purpose of these test cases is to verify that bundles are transmitted using LTP Green when the ECOS Streaming bit is set and LTP Red otherwise. Test case OTH.g originates bundles in LTP segments with an ION node and a DTN2 node must disposition the segment and bundles. Test case OTH.h has a DTN2 node originating bundles in LTP segments and an ION node dispositioning the segment and bundles. Tests were conducted without custody transfer. Logs were retrieved after each activity for analysis.

The data flow is depicted in Figure 6-10 and the node configuration is detailed in Table 6-10.

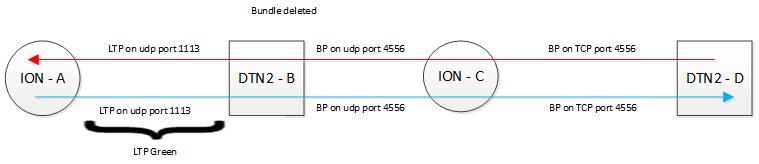


Figure A-13: LTP Green Test Case OTH.g and OTH.h - Data Flow

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| Node A – ION | ipn:17000.0 | j.j.j.200 | ltp / 1113 | ltp 19000 n.n.n.4:1113 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | n.n.n.4 | ltp / 1113  udp / 4556 | ltp 17000 j.j.j.200:1113  udp 20000 j.j.j.220:4556 1443 | 21000 21000 ipn:20000.0 |
| Node C – ION | ipn:20000.0 | j.j.j.220 | udp / 4556  tcp / 4556 | udp 19000 n.n.n.4:4556 1400  tcp 21000 n.n.n.6:4556 | 17000 17000 ipn:19000.0 |
| Node D – DTN2 | ipn:21000.0 | n.n.n.6 | tcp / 4556 | tcp 20000 j.j.j.220:4556 | 17000 17000 ipn:20000.0 |

Table A-31: LTP Green Test Case OTH.g and OTH.h – Node Configuration

Expected Results

1. Users at Nodes A and D will initiate tests. Test can be conducted simultaneously.
2. Only green LTP segments will be transmitted.
3. LTP segments will be forwarded to adjacent routers prior to the SDA timeout .
4. Bundles will be forwarded when de-encapsulated.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.g-1 | Start all 4 DTN nodes |  | Success |
| OTH.g-2 | Start wireshark or tcpdump capture between Nodes A and B |  | Success |
| OTH.g-3 | Node D:  dtnsink –p –v -n 10 ipn:21000.2 | Prepare to receive 10 bundles | Success |
| OTH.g-4 | Node A:  dtnperf\_vION --client –r –f --del  –l 300 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 1b –D 5k | Send 5 bundles with 1000 byte payload at a rate of 1 bundle per second. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (LTP Red should be used) | Success |
| OTH.g-5 | Node A:  dtnperf\_vION --client –r –f --del  –l 300 -m ipn:17000.0  -d ipn:21000.2  –P 1k –R 1b –D 5k --unreliable | Send 5 bundles with 1000 byte payload at a rate of 1 bundle per second. Status reports will be sent to ipn:17000.0 and logged to ion.log.  (LTP Green should be used) | Success |
| OTH.g-6 | Node D: Monitor dtnsink output | A minimum of 5 bundles and up to 10 bundles should be received | Success |
| OTH.g-7 | Stop network capture(s) and examine | First 5 bundles should have been transmitted using LTP Red and the next 5 should be transmitted using LTP Green | Success |
| OTH.g-8 | Save log files, etc. |  | Success |

Table A-32: Test Cases OTH.g

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.h-1 | Start all 4 DTN nodes |  | Success |
| OTH.h-2 | Start wireshark or tcpdump capture between Nodes A and B |  | Success |
| OTH.h-3 | Node A:  bpsink ipn:17000.2 | Prepare to receive 10 bundles | Success |
| OTH.h-4 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 300 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 1b –D 5k | Send 5 bundles without the ECOS streaming bit set. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (LTP Red should be used) | Success |
| OTH.h-5 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 300 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 1b –D 5k --unreliable | Send 5 bundles with the ECOS streaming bit set. Status reports will be sent to ipn:21000.0 and logged to dtn.log.  (LTP Green should be used) | Success |
| OTH.h-6 | Node A: Monitor bpsink output | A minimum of 5 bundles and up to 10 bundles should be received | Success |
| OTH.h-7 | Stop network capture(s) and examine | First 5 bundles should have been transmitted using LTP Red and the next 5 should be transmitted using LTP Green | Success |
| OTH.h-8 | Save log files, etc. |  | Success |

Table A-33: Test Cases OTH.h

* + - 1. ECOS Critical bit set cases OTH.i and OTH.j

The purpose of these test cases is to verify that ECOS Critical bit set results in transmission over all plausible routes. This was accomplished by setting bit 0x01 of the ECOS block’s flags byte to 1. These test cases will utilize four routers in a diamond topology. OTH.i has ION node A as an originating source and OTH.j has a DTN2 router as the originating source. The test were conducted when all paths are unobstructed. When behavior has been verified for test case OTH.i, path between Node B and D were broken and the test re-executed. The path between Node B and D was re-established and the path between Node C and D was broken and the test re-executed.

Test case OTH.j was conducted in the reverse direction and in the same manner. The test was conducted when all paths are unobstructed. When behavior has been verified for test case OTH.j, the path between Node B and A was broken and the test re-executed. The path between Node B and A was re-established and the path between Node C and A was broken and the test re-executed.

Due to the possibility of flooding the VPN with bundles, these test cases will not be conducted across the internet over a VPN. During the upcoming CCSDS conference in March, accommodations are being made to test at CalTech in Pasedena, California, USA. MSFC/HOSC provided a 3Com OfficeConnect 8 port hub as well as two laptops to host DTN2 routers. JAXA providied supporting hardware for two ION nodes. Nodes provided by each team will comply with configurations of test cases OTH.i and OTH.j .

The data flow is depicted in Figure 6-11 and the node configuration is detailed in Table 6-11.

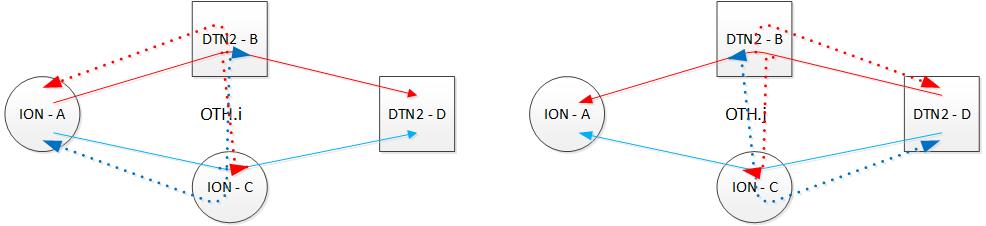


Figure A-14: ECOS Critical bit set cases OTH.i and OTH.j - Data Flow

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Node and**  **Implementation** | **IPN Scheme EID** | **IP Address** | **Induct / Port** | **Outduct**  **(ION “add outduct” format)** | **Group Routes** |
| Node A – ION | ipn:17000.0 | x.x.x.1 | ltp / 1113  udp / 4556 | ltp 19000 x.x.x.2:1113  udp 20000 x.x.x.3:4556 1350 | 21000 21000 ipn:19000.0 |
| Node B – DTN2 | ipn:19000.0 | x.x.x.2 | ltp / 1113  udp / 4556  tcp / 4556 | ltp 17000 x.x.x.1:1113  udp 20000 x.x.x.3:4556  tcp 21000 x.x.x.4:4556 |  |
| Node C – ION | ipn:20000.0 | x.x.x.3 | udp / 4556  tcp / 4556 | udp 17000 x.x.x.1:4556 1350  udp 19000 x.x.x.2:4556 1350  tcp 21000 x.x.x.4:4556 |  |
| Node D – DTN2 | ipn:21000.0 | x.x.x.4 | tcp / 4556 | tcp 20000 x.x.x.3:4556  tcp 19000 x.x.x.2:4556 | 17000 17000 ipn:20000.0 |

Table A-34: ECOS Critical bit set cases OTH.i and OTH.j – Node Configuration

Expected Results

1. Users at Nodes A and D will initiate tests. Tests cannot be conducted simultaneously.
2. The first execution of a test case, the bundle will arrive at the remote destination twice.
3. For the second and third executions of a test case, only one bundle will arrive at the remote destination
4. A copy of each bundle may be returned to the source and if so should not be re-forwarded

Additional Special Configuration

Nodes B and D will utilize a NASA developed DTN2 External Router during this test as the built in static router does not currently support ECOS priority routing.

Test Procedures

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.i-1 | Start all 4 DTN nodes |  | SUCCESS |
| OTH.i-2 | Start wireshark or tcpdump capture on Nodes A and D |  | SUCCESS |
| OTH.i-3 | Node D:  dtnsink –p –v –n 100 ipn:21000.2 | Prepare to receive bundles | SUCCESS |
| OTH.i-4 | Node A:  bptrace ipn:17000.1 ipn:21000.2 ipn:17000.0 60 0.1  ‘normal message’ rcv,ct,fwd,dlv.del | Send 1 normal priority bundle with lifetime 60 seconds to Node D. Status reports will be sent to ipn:17000.0 and logged to ion.log. | SUCCESS |
| OTH.i-5 | Node D: Monitor dtnsink output and dtn.log | Verify only 1 bundle received and it came from Node B | SUCCESS |
| OTH.i-6 | Node A:  bptrace ipn:17000.1 ipn:21000.2 ipn:17000.0 60 0.1.0.0.1  ‘priority message’ | Send 1 bundle with critical ECOS priority to Node D. | SUCCESS |
| OTH.i-7 | Node D: Monitor dtnsink output and dtn.log | Verify 1 bundle received from both Node B and Node C recognizing that only 1 may be delivered to dtnsink due to duplicate delivery suppression. | SUCCESS |
| OTH.i-8 | Node D:  > Shutdown CL to Node C | Only route from C to D blocked | SUCCESS |
| OTH.i-9 | Node A:  bptrace ipn:17000.1 ipn:21000.2 ipn:17000.0 60 0.1.0.0.1  ‘priority message’ | Send 1 bundle with critical ECOS priority to Node D. | SUCCESS |
| OTH.i-10 | Node D: Monitor dtnsink output and dtn.log | Verify 1 bundle received from Node B. | SUCCESS |
| OTH.i-11 | Node D:  > shutdown CL to Node B  > re-open CL to Node C | Only route from B to D blocked | SUCCESS |
| OTH.i-12 | Node A:  bptrace ipn:17000.1 ipn:21000.2 ipn:17000.0 60 0.1.0.0.1  ‘priority message’ | Send 1 bundle with critical ECOS priority to Node D. | SUCCESS |
| OTH.i-13 | Node D: Monitor dtnsink output and dtn.log | Verify 1 bundle received from Node C. | SUCCESS |
| OTH.i-14 | Stop network capture(s) and examine | Verify expected transmissions | SUCCESS |
| OTH.i-15 | Node A: Examine ion.log | Status reports should have been logged | SUCCESS |
| OTH.i-16 | Save log files, etc. |  | SUCCESS |

**Table A-32: Test Cases OTH.i**

| **Step** | **Step Description** | **Comment / Expected Result** | **Success /**  **Fail** |
| --- | --- | --- | --- |
| OTH.j-1 | Start all 4 DTN nodes |  | Success |
| OTH.j-2 | Start ireshark or tcpdump capture on Nodes A and D |  | Success |
| OTH.j-3 | Node A:  bpsink ipn:17000.2 | Prepare to receive bundles | Success |
| OTH.j-4 | Node D:  dtnperf\_vDTN2 –client –r –f –del  --force-eid IPN –ipn-local 21000  –l 3600 –m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k | Send 1 normal priority bundle with lifetime 3600 seconds to Node A. Status reports will be sent to ipn:21000.0 and logged to dtn.log. | Success |
| OTH.j-5 | Node A: Monitor bpsink output and ion.log | Verify only 1 bundle received and it came from Node C | Success |
| OTH.j-6 | Node D:  dtnperf\_vDTN2 –client –r –f –del  --force-eid IPN –ipn-local 21000  –l 3600 –m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k –critical | Send 1 bundle with critical ECOS priority to Node A. | Success |
| OTH.j-7 | Node A: Monitor bpsink output and ion.log | Verify 1 bundle received from both Node B and Node C recognizing that only 1 may be delivered to bpsink due to duplicate delivery suppression. | Success |
| OTH.j-8 | Node A:  > Shutdown CL to Node C | Only route from C to A blocked | Success |
| OTH.j-9 | Node D:  dtnperf\_vDTN2 –client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k --critical | Send 1 bundle with critical ECOS priority to Node D. | Success |
| OTH.j-10 | Node A: Monitor bpsink output and ion.log | Verify 1 bundle received from Node B. | Success |
| OTH.j-11 | Node A:  > shutdown CL to Node B  > re-open CL to Node C | Only route from B to A blocked | Success |
| OTH.j-12 | Node D:  dtnperf\_vDTN2 --client –r –f --del  --force-eid IPN --ipn-local 21000  –l 3600 -m ipn:21000.0  -d ipn:17000.2  –P 1k –R 20b –D 1k --critical | Send 1 bundle with critical ECOS priority to Node D. | Success |
| OTH.j-13 | Node A: Monitor bpsink output and ion.log | Verify 1 bundle received from Node C. | Success |
| OTH.j-14 | Stop network capture(s) and examine | Verify expected transmissions | Success |
| OTH.j-15 | Node D: Examine dtn.log | Status reports should have been logged | Success |
| OTH.j-16 | Save log files, etc. |  | Success |

**Table A-32: Test Cases OTH.j**

1. : Delay Tolerant Payload Conditioning (DTPC) Interoperability Testing

The Delay Tolerant Payload Conditioning (DTPC) implementation for the DTN2 Bundle Protocol implementation was provided by the TopCoder organization. As part of the implementation, TopCoder performed interoperability tests between their implementation and ION. The results of those tests are documented here.

In the report below, tests 7, 14, and 21 were not performed. These tests do NOT impact interoperability because they test functions entirely internal to an implementation.

Tests 16 and 19 were not performed, but their peers, tests 9 and 12, were. Tests 9 and 12 demonstrate the interoperability between the two different implementations in one direction and are enough to confirm interoperability for the purposes of CCSDS interoperability testing.

* + 1. Test report

As the test results, all test suites passed.

The test suites are arranged as below:

| **Test suite** | **Purpose** | **Group** | **Success / Fail** |
| --- | --- | --- | --- |
| test1 | Delivery of application data items in transmission (rather than reception) order | Dtn2 -> Dtn2 | Success |
| test2 | Detection of reception gaps in the sequence of transmitted application data items | Dtn2 -> Dtn2 | Success |
| test3 | End-to-end positive acknowledgment of successfully received data | Dtn2 -> Dtn2 | Success |
| test4 | End-to-end retransmission of missing data, driven by timer expiration | Dtn2 -> Dtn2 | Success |
| test5 | Suppression of duplicate application data items | Dtn2 -> Dtn2 | Success |
| test6 | Aggregation of small application data items into large bundle payloads, to reduce bundle protocol overhead | Dtn2 -> Dtn2 | Success |
| test7 | Application-controlled elision of redundant data items in aggregated payloads, to improve link utilization | Dtn2 -> Dtn2 | CANCELED |
| test8 | Delivery of application data items in transmission (rather than reception) order | ION -> Dtn2 | Success |
| test9 | Detection of reception gaps in the sequence of transmitted application data items | ION -> Dtn2 | Success |
| test10 | End-to-end positive acknowledgment of successfully received data | ION -> Dtn2 | Success |
| test11 | End-to-end retransmission of missing data, driven by timer expiration | ION -> Dtn2 | Success |
| test12 | Suppression of duplicate application data items | ION -> Dtn2 | Success |
| test13 | Aggregation of small application data items into large bundle payloads, to reduce bundle protocol overhead | ION -> Dtn2 | Success |
| test14 | Application-controlled elision of redundant data items in aggregated payloads, to improve link utilization | ION -> Dtn2 | CANCELED |
| test15 | Delivery of application data items in transmission (rather than reception) order | Dtn2 -> ION | Success |
| test16 | Detection of reception gaps in the sequence of transmitted application data items | Dtn2 -> ION | CANCELED |
| test17 | End-to-end positive acknowledgment of successfully received data | Dtn2 -> ION | Success |
| test18 | End-to-end retransmission of missing data, driven by timer expiration | Dtn2 -> ION | Success |
| test19 | Suppression of duplicate application data items | Dtn2 -> ION | CANCELED |
| test20 | Aggregation of small application data items into large bundle payloads, to reduce bundle protocol overhead | Dtn2 -> ION | Success |
| test21 | Application-controlled elision of redundant data items in aggregated payloads, to improve link utilization | Dtn2 -> ION | CANCELED |

Note that In the form above, test7/14/21 are canceled.

And tests16/19 are not at the DTN2 side and ION does NOT have glue to verify them. So they are also canceled.

* 1. Resource Contact List

|  |  |
| --- | --- |
| **Name** | **Resource Email** |
| caoweiquan322 | caoweiquan322@126.com |