

## Recommendation for Space Data System Standards

**LICKLIDER  
TRANSMISSION  
PROTOCOL (LTP) FOR  
CCSDS**

**DRAFT RECOMMENDED STANDARD**

**CCSDS 734.1-B-0**

**DRAFT BLUE BOOK**

**February 2015**

## **DEDICATION**

This book is dedicated to Adrian Hooke, whose end-to-end sensibilities and tireless advocacy for standardization of space data systems directly contributed to the formation of the Consultative Committee for Space Data Systems in 1982. His unique combination of technical skill, management abilities, and vision served CCSDS well for over 30 years. During that time CCSDS solidified the standardization of Physical and Data Link Layer protocols, and developed standards and technologies that had important and wide-ranging impacts in both the space and terrestrial communications industries. In the late 1990s, Adrian envisioned a new era for space communications leveraging a confluence of terrestrial internetworking and space-based data transport technologies. This led to the development of a concept that has come to be known as the Solar System Internetwork (SSI), of which the Licklider Transmission Protocol described here is a part.

Adrian will be missed, by CCSDS for the scope of his technical contributions and his leadership, and by his colleagues and friends for the greatness of his spirit and his wit. But his legacy to the space community remains. CCSDS will continue to provide useful and innovative solutions to space communication challenges so that Adrian's vision of an interoperable, standards-based communication system that reduces mission development time, cost, and risk will eventually be realized.

## **AUTHORITY**

Issue:	Draft Recommended Standard, Issue 0
Date:	February 2015
Location:	Washington, DC, USA

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4), and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the e-mail address below.

This document is published and maintained by:

CCSDS Secretariat  
National Aeronautics and Space Administration  
Washington, DC, USA  
E-mail: [secretariat@mailman.ccsds.org](mailto:secretariat@mailman.ccsds.org)

## **STATEMENT OF INTENT**

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of its members. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommended Standards** and are not considered binding on any Agency.

This **Recommended Standard** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommendation** is entirely voluntary. Endorsement, however, indicates the following understandings:

- o Whenever a member establishes a CCSDS-related **standard**, this **standard** will be in accord with the relevant **Recommended Standard**. Establishing such a **standard** does not preclude other provisions which a member may develop.
- o Whenever a member establishes a CCSDS-related **standard**, that member will provide other CCSDS members with the following information:
  - The **standard** itself.
  - The anticipated date of initial operational capability.
  - The anticipated duration of operational service.
- o Specific service arrangements shall be made via memoranda of agreement. Neither this **Recommended Standard** nor any ensuing **standard** is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this **Recommended Standard** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or (3) be retired or canceled.

In those instances when a new version of a **Recommended Standard** is issued, existing CCSDS-related member standards and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each member to determine when such standards or implementations are to be modified. Each member is, however, strongly encouraged to direct planning for its new standards and implementations towards the later version of the Recommended Standard.

## **FOREWORD**

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CCSDS has processes for identifying patent issues and for securing from the patent holder agreement that all licensing policies are reasonable and non-discriminatory. However, CCSDS does not have a patent law staff, and CCSDS shall not be held responsible for identifying any or all such patent rights.

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Standard is therefore subject to CCSDS document management and change control procedures, which are defined in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4). 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 sent to the CCSDS Secretariat at the e-mail 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 (BFSP0)/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.
- Electronics and Telecommunications Research Institute (ETRI)/Korea.
- 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**

<b>Document</b>	<b>Title</b>	<b>Date</b>	<b>Status</b>
CCSDS 734.1-B-0	Licklider Transmission Protocol (LTP) for CCSDS, Draft Recommended Standard, Issue 0	February 2015	Current draft

## CONTENTS

<u>Section</u>	<u>Page</u>
<b>1 INTRODUCTION.....</b>	<b>1-1</b>
1.1 PURPOSE.....	1-1
1.2 SCOPE.....	1-1
1.3 ORGANIZATION OF THE DOCUMENT.....	1-1
1.4 CONVENTIONS AND DEFINITIONS.....	1-2
1.5 NOMENCLATURE .....	1-5
1.6 REFERENCES .....	1-6
<b>2 OVERVIEW.....</b>	<b>2-1</b>
2.1 GENERAL.....	2-1
2.2 ARCHITECTURAL ELEMENTS .....	2-2
2.3 SERVICE PROVIDED BY LTP.....	2-2
<b>3 CCSDS PROFILE OF RFC 5326.....</b>	<b>3-1</b>
3.1 BASE SPECIFICATIONS .....	3-1
3.2 AMBIGUITY RESOLUTION .....	3-1
3.3 LTP OVER UDP .....	3-1
3.4 LTP FOR CCSDS.....	3-1
3.5 LIMITS ON THE RANGES OF LTP FIELD VALUES .....	3-2
3.6 AGENCY USE OF LTP ENGINE IDS.....	3-3
3.7 GREEN-PART DATA .....	3-4
3.8 LTP EXTENSIONS.....	3-4
3.9 LTP SECURITY.....	3-4
<b>4 LTP SERVICE SPECIFICATION.....</b>	<b>4-1</b>
4.1 SERVICES AT THE USER INTERFACE .....	4-1
4.2 SUMMARY OF PRIMITIVES.....	4-1
4.3 SUMMARY OF PARAMETERS .....	4-2
4.4 LTP SERVICE PRIMITIVES .....	4-4
<b>5 SERVICES LTP REQUIRES OF THE SYSTEM.....</b>	<b>5-1</b>
5.1 RELIABLE STORAGE SERVICE.....	5-1
5.2 UNDERLYING COMMUNICATION SERVICE REQUIREMENTS .....	5-1



## CONTENTS (continued)

<u>Section</u>	<u>Page</u>
<b>6 CONFORMANCE REQUIREMENTS .....</b>	<b>6-1</b>
6.1 PICS PROFORMA .....	6-1
6.2 LICKLIDER TRANSMISSION PROTOCOL REQUIREMENTS .....	6-1
<b>7 CLIENT OPERATIONS.....</b>	<b>7-1</b>
7.1 OVERVIEW—LTP SERVICE DATA AGGREGATION (SDA).....	7-1
7.2 LTP SDA SPECIFICATION.....	7-1
<b>ANNEX A USING THE CCSDS SPACE PACKET OR ENCAPSULATION SERVICE AS AN UNDERLYING COMMUNICATION SERVICE FOR LTP (NORMATIVE).....</b>	<b>A-1</b>
<b>ANNEX B LICKLIDER TRANSMISSION PROTOCOL MANAGEMENT INFORMATION BASE (NORMATIVE).....</b>	<b>B-1</b>
<b>ANNEX C PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT PROFORMA (NORMATIVE).....</b>	<b>C-1</b>
<b>ANNEX D SECURITY, SANA, AND PATENT CONSIDERATIONS (INFORMATIVE) .....</b>	<b>D-1</b>
<b>ANNEX E INFORMATIVE REFERENCES (INFORMATIVE) .....</b>	<b>E-1</b>
<b>ANNEX F ACRONYMS AND ABBREVIATIONS (INFORMATIVE).....</b>	<b>F-1</b>

### Figure

1-1 LTP’s Relationship to Neighboring Protocols.....	1-3
2-1 Protocol Stack View of LTP Architectural Elements.....	2-2
2-2 Communications View of LTP .....	2-3
2-3 Overview of LTP Interactions .....	2-4

### Table

B-1 Local Engine Configuration Information .....	B-2
B-2 Remote Engine Configuration Information .....	B-3
C-1 PICS Notation .....	C-2
C-2 PICS Conditional Status Notation .....	C-3
C-3 Symbols for PICS ‘Protocol Feature’ Column .....	C-3
C-4 Symbols for PICS ‘Support’ Column .....	C-3
D-1 Initial CCSDS LTP Engine ID Registry .....	D-4
D-2 Initial CCSDS LTP Client Service ID Number Registry .....	D-5

## **1 INTRODUCTION**

### **1.1 PURPOSE**

This document defines a Recommended Standard for the CCSDS Licklider Transmission Protocol (LTP) and associated service for application in the space environment. LTP provides optional reliability mechanisms on top of an underlying (usually data link) communication service.

### **1.2 SCOPE**

LTP is intended for use over the current and envisaged packet delivery services used in the space environment, including:

- CCSDS conventional packet telecommand;
- CCSDS conventional packet telemetry.

For space data links, LTP will typically be deployed over a CCSDS data link that supports CCSDS Encapsulation Packets so that one LTP segment can be encapsulated in a single Encapsulation packet. LTP may also operate over a wide variety of ground-network services including those specified by the CCSDS for cross-support purposes.

### **1.3 ORGANIZATION OF THE DOCUMENT**

This Recommended Standard is organized as follows:

- a) Section 2 contains a descriptive overview of LTP operation as well as a brief history of the protocol's heritage. Users not already familiar with LTP may want to start with this section.
- b) Section 3 contains a profile of RFC 5326 (reference [2]) for use by CCSDS.
- c) Section 4 contains the abstract service specification for LTP.
- d) Section 5 specifies the services that LTP requires from the underlying system.
- e) Section 6 contains conformance requirements for the CCSDS profile of LTP.
- f) Section 7 defines a client operations service that allows multiple layer-(N+1) SDUs to be aggregated into a single LTP block in order to improve efficiency.
- g) Annex A specifies how to layer LTP over the CCSDS Space Packet Service or the CCSDS Encapsulation Service.
- h) Annex B contains the Management Information Base (MIB) for the protocol.
- i) Annex C contains the Protocol Implementation Conformance Statement (PICS) Proforma.

- j) Annex D discusses security, SANA, and patents considerations related to the specification.
- k) Annex E is a list of informative references.
- l) Annex F is a list of abbreviations and acronyms that appear in the document.

## 1.4 CONVENTIONS AND DEFINITIONS

### 1.4.1 TERMS

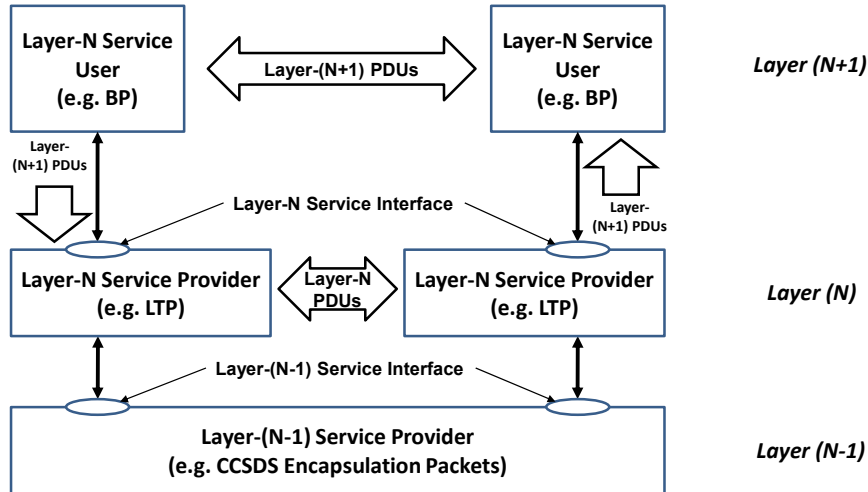
#### 1.4.1.1 Definitions from OSI Basic Reference Model

This Recommended Standard makes use of a number of terms defined in reference [1]. The use of those terms in this Recommended Standard shall be understood in a generic sense, i.e., in the sense that those terms are generally applicable to any of a variety of technologies that provide for the exchange of information between real systems. Those terms are:

- entity;
- Protocol Data Unit (PDU);
- service;
- Service Access Point (SAP);
- Service Data Unit (SDU).

Figure 1-1 illustrates the relationship of the LTP protocol defined in this document and protocols at the layers above and below LTP. From the point of view of protocols above LTP (e.g., Bundle Protocol), the service LTP provides is optionally reliable delivery of layer-(N+1) PDUs across a link (represented by the dotted lines in the figure). From LTP's point of view, **delivery across a particular data link** is provided by a layer-(N-1) service such as the **CCSDS Encapsulation Service**.

Figure 1-1 illustrates the general service user-service provider relationships among layers. For the specific case of LTP in the CCSDS stack, the LTP service sits between the Data Link Layer and the Network Layer.



**Figure 1-1: LTP's Relationship to Neighboring Protocols**

#### 1.4.1.2 Definitions from Open Systems Interconnection (OSI) Service Definition Conventions

This Recommended Standard makes use of a number of terms defined in reference [1]. The use of those terms in this Recommended Standard shall be understood in a generic sense, i.e., in the sense that those terms are generally applicable to any of a variety of technologies that provide for the exchange of information between real systems. Those terms are:

- indication;
- primitive;
- request;
- response.

#### 1.4.1.3 Definitions from RFC 5326

This Recommended Standard makes use of a number of terms defined in reference [2]. Some of the definitions needed for section 2 of this document are reproduced here for convenience.

**engine ID:** An integer that uniquely identifies a given LTP engine, within some closed set of communicating LTP engines.

NOTE – When LTP is operating underneath the Delay-Tolerant Networking (DTN) Bundle Protocol (BP), the convergence layer adapter mediating the two will be responsible for translating between DTN endpoint IDs and LTP engine IDs in an implementation-specific manner.

**block:** An array of contiguous octets of application data handed down by the upper layer protocol (typically BP) to be transmitted from one LTP client service instance to another.

Any subset of a block comprising contiguous octets beginning at the start of the block is termed a 'block prefix', and any such subset of the block ending with the end of the block is termed a 'block suffix'.

**red-part:** The block prefix that is to be transmitted reliably, i.e., subject to acknowledgment and retransmission.

**green-part:** The block suffix that is to be transmitted unreliably, i.e., not subject to acknowledgments or retransmissions. If present, the green-part of a block begins at the octet following the end of the red-part.

**session:** A thread of LTP protocol activity conducted between two peer engines for the purpose of transmitting a block. Data flow in a session is unidirectional: data traffic flows from the sending peer to the receiving peer, while data-acknowledgment traffic flows from the receiving peer to the sending peer.

**sender:** The data-sending peer of a session.

**receiver:** The data-receiving peer of a session.

**client service instance:** A software entity, such as an application or a higher-layer protocol implementation, that is using LTP to transfer data.

**segment:** The unit of LTP data transmission activity. It is the data structure transmitted from one LTP engine to another in the course of a session. Each LTP segment is of one of the following types: data segment, report segment, report-acknowledgment segment, cancel segment, cancel-acknowledgment segment.

**End Of Block (EOB):** The last data segment transmitted as part of the original transmission of a block. This data segment also indicates that the segment's upper bound is the total length of the block (in octets).

**End Of Red-Part (EORP):** The segment transmitted as part of the original transmission of a block containing the last octet of the block's red-part. This data segment also indicates that the segment's upper bound is the length of the block's red-part (in octets).

**checkpoint:** A data segment soliciting a reception report from the receiving LTP engine. The EORP segment must be flagged as a checkpoint, as must the last segment of any retransmission; these are 'mandatory checkpoints'. All other checkpoints are 'discretionary checkpoints'.

**client service ID:** The client service ID number identifies the upper-level service to which the segment is to be delivered by the receiver. It is functionally analogous to a TCP port number. If multiple instances of the client service are present at the destination, multiplexing

must be done by the client service itself on the basis of information encoded within the transmitted block.

**Self-Delimiting Numeric Value (SDNV):** A representation of integer values in binary format where the length of the representation is a function of the value being represented.

NOTE – Definition (20) of RFC 5326 (reference [2]) or RFC 6256 (reference [E5]) can be consulted for additional explanation and examples.

#### **1.4.1.4 Other Definitions**

**Application Process Identifier (APID):** Part of the path ID used to identify a logical data path for CCSDS Space Packets.

**Underlying Communication Protocols (UCP):** The communication protocols used by LTP to transfer segments between LTP engines.

## **1.5 NOMENCLATURE**

### **1.5.1 NORMATIVE TEXT**

The following conventions apply for the normative specifications in this Recommended Standard:

- a) the words ‘shall’ and ‘must’ imply a binding and verifiable specification;
- b) the word ‘should’ implies an optional, but desirable, specification;
- c) the word ‘may’ implies an optional specification;
- d) the words ‘is’, ‘are’, and ‘will’ imply statements of fact.

NOTE – These conventions do not imply constraints on diction in text that is clearly informative in nature.



### **1.5.2 INFORMATIVE TEXT**

In the normative sections of this document, informative text is set off from the normative specifications either in notes or under one of the following subsection headings:

- Overview;
- Background;
- Rationale;
- Discussion.

## 1.6 REFERENCES

The following documents contain provisions which, through reference in this text, constitute provisions of this Recommended Standard. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Recommended Standard are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS Recommended Standards.

- [1] *Information Technology—Open Systems Interconnection—Basic Reference Model—Conventions for the Definition of OSI Services*. International Standard, ISO/IEC 10731:1994. Geneva: ISO, 1994.
- [2] M. Ramadas, S. Burleigh, and S. Farrell. *Licklider Transmission Protocol—Specification*. RFC 5326. Reston, Virginia: ISOC, September 2008.
- [3] S. Farrell, M. Ramadas, and S. Burleigh. *Licklider Transmission Protocol—Security Extensions*. RFC 5327. Reston, Virginia: ISOC, September 2008.
- [4] “Port Numbers.” Internet Assigned Numbers Authority. Internet Corporation for Assigned Names and Numbers. <http://www.iana.org/assignments/port-numbers>.
-  [5] “Licklider Transmission Protocol Engine Identifiers.” Space Assigned Numbers Authority. [http://sanaregistry.org/r/ltp\\_engineid/](http://sanaregistry.org/r/ltp_engineid/).
-  [6] “Licklider Transmission Protocol (LTP) Parameters.” Internet Assigned Numbers Authority. Internet Corporation for Assigned Names and Numbers. <http://www.iana.org/assignments/ltp-parameters/ltp-parameters.xhtml>.
- [7] *Encapsulation Service*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.1-B-2. Washington, D.C.: CCSDS, October 2009.
- [8] “Protocol Identifier for Encapsulation Service.” Space Assigned Numbers Authority. [http://sanaregistry.org/r/protocol\\_id/](http://sanaregistry.org/r/protocol_id/).
- [9] *Space Packet Protocol*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.0-B-1. Washington, D.C.: CCSDS, September 2003.
- [10] “Space Packet Protocol Application Process Identifier (APID).” Space Assigned Numbers Authority. [http://sanaregistry.org/r/space\\_packet\\_protocol\\_application\\_process\\_id/](http://sanaregistry.org/r/space_packet_protocol_application_process_id/).

## 2 OVERVIEW

### 2.1 GENERAL

CCSDS has identified requirements for a protocol to sit between an internetworking protocol such as the Bundle Protocol (reference [E1]) and the various CCSDS data links (see reference [E3]). The two requirements identified in reference [E3] for such a layer-N protocol are reliable delivery of layer-(N+1) PDUs and the ability to aggregate multiple layer (N+1) PDUs into a single layer-N PDU for the purposes of reliable delivery across the link.

Reliable data delivery is accomplished by the Red-Part delivery service of the LTP protocol described in section 3 of this document. Aggregation of multiple layer-(N+1) SDUs into a single layer-N PDU (LTP block) is achieved by the implementation of the standardized 'service data aggregation' client operation described in section 7. Such an adaptor is envisaged by figure 2-4 of reference [E3], which lists an 'LTP/ENCAP Convergence Layer Adaptor' as a work item. The rationale for aggregating multiple layer-(N+1) PDUs into a single layer-N PDU for the purposes of reliable delivery is that it may allow the system to reduce the acknowledgement-channel bandwidth in the case that the layer-(N+1) (and higher) protocols transmit many small PDUs, each of which might otherwise require independent acknowledgement.

The Licklider Transmission Protocol (LTP) sits between the Data Link and the Network Layers of the ISO stack and provides **optionally reliable communications over a single data link hop**. While LTP can be deployed over multi-hop services (e.g., UDP) on the ground, this document recommends that LTP be terminated at the ends of each long-delay, error-prone, or disruption-prone link (such as a space link) in what might be a multi-hop path. Thus when considering LTP's suitability for use on space links, it is enough to consider its functionality, its scalability to multiple 1-hop peers, and its interaction with other protocols on a single space link. The main restriction on LTP's scalability to multiple peers is the storage required to maintain data for retransmission between contacts to a particular peer. Because of the sparseness of space communications, it is not envisioned that this scalability with the number of peers will be an issue.

LTP was originally developed for space communication and is largely derived from the Acknowledged-mode procedures of the CCSDS File Delivery Protocol (CFDP). The protocol specification below is a reproduction of the Internet Engineering Task Force (IETF) Request for Comments (RFC) 5326, Licklider Protocol Specification. A separate motivational RFC (reference [E2]) provides the motivation behind the IETF specification of the protocol and may be informative in this context.

The IETF's classification of the Licklider Transmission Protocol RFC as 'experimental' should be considered in the context of LTP's deployment on the global Internet, where millions of end systems are exchanging millions of data flows at any given instant, and where protocols such as LTP are typically thought of as 'transport' or 'end-to-end' protocols operating across multiple data links. In the Internet context, issues such as scalability to millions of nodes, congestion control, and non-destructive coexistence with other established protocols (in particular the Transmission Control Protocol [TCP]), are of extreme



importance. The specification's status as 'experimental' for use in the Internet is independent of and orthogonal to its applicability for space use. As stated above, because this document recommends LTP for use over individual space links, the scalability concerns associated with LTP deployment in the Internet do not pertain to CCSDS.

While LTP could be used as a bearer mechanism for cross-support between CCSDS Agencies across the Internet, it is more likely that Internet protocols such as the TCP mentioned above would be used for that purpose, in part because of the issues that cause LTP to be marked as 'experimental' for use in the Internet.

## 2.2 ARCHITECTURAL ELEMENTS

The architectural elements of LTP are depicted in figure 2-1.

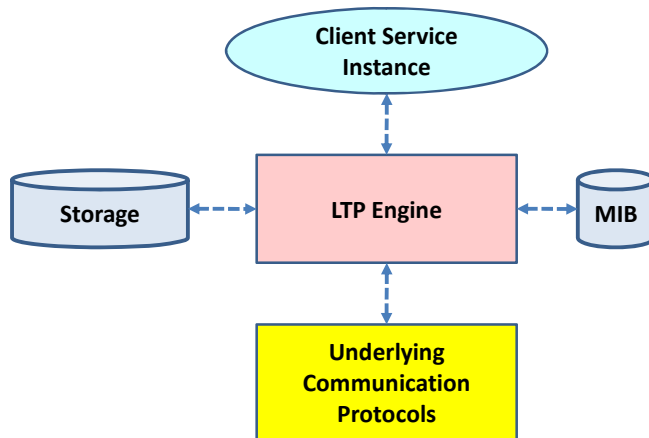


Figure 2-1: Protocol Stack View of LTP Architectural Elements

## 2.3 SERVICE PROVIDED BY LTP

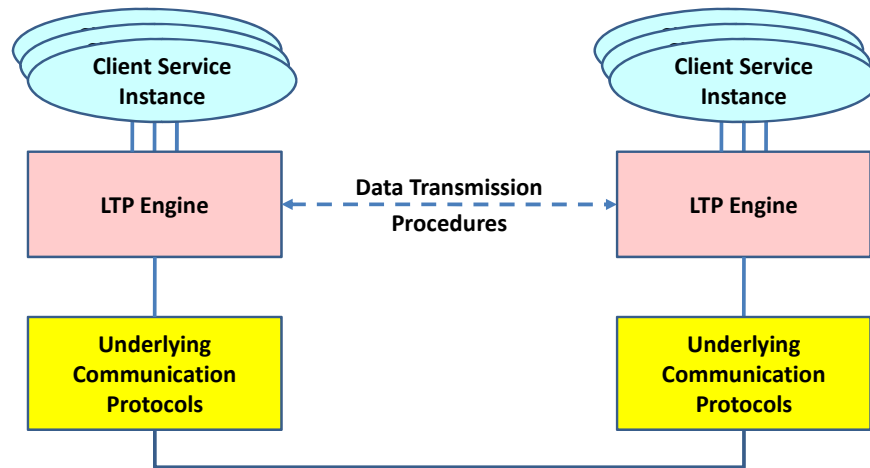
### 2.3.1 GENERAL

LTP provides a data transmission service to move blocks of data from one LTP engine to another, where in general the two engines are resident in separate data systems, often with a single connecting space link.

Each block consists logically of two parts, either of which may be of length zero. The first part, termed the 'red-part', is transmitted reliably between LTP entities, using acknowledgements and retransmissions to ensure that the entire red-part is received reliably at the receiver; this provides a reliable transmission service. The second part of the block, termed the 'green-part', consists of data to be transmitted unreliably. Data in the green-part is not subject to acknowledgements and retransmissions and therefore provides an unreliable service. The LTP client Service Instance controls what data in a block is 'red' and what is

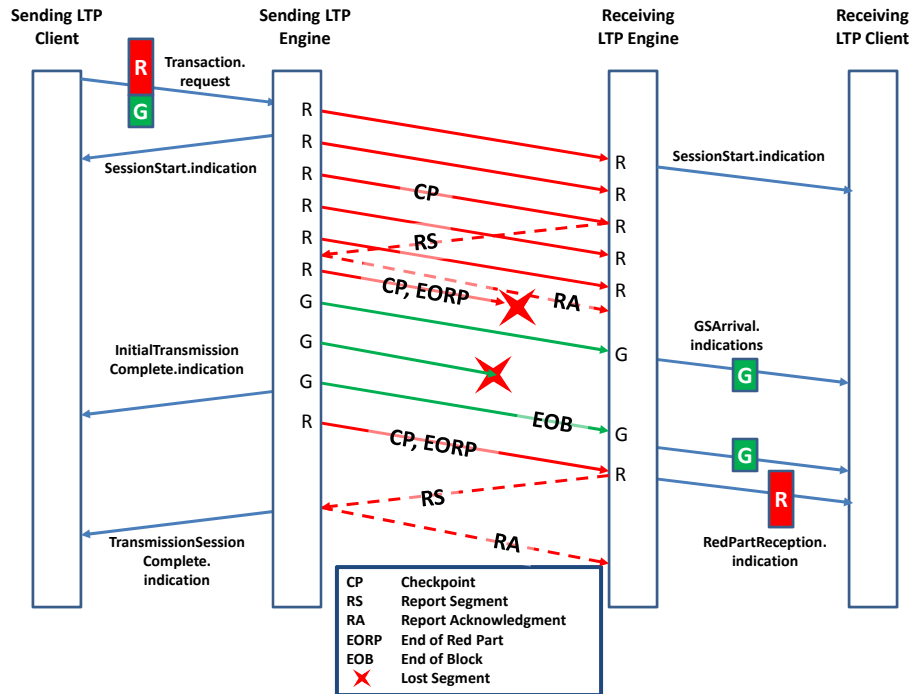
‘green’. A Client Service Instance that desires completely reliable data transfer must therefore specify that all of the data be sent as ‘red’ (reliable) data. In this specification, the ability to send/receive green-part data is optional. However, if green-part capability is supported, then both transmission and reception must be supported. Block transmission may span periods of disconnection. During these periods, retransmission timers maintained by LTP are suspended.

As depicted in figure 2-2, below, the data transmission procedures constitute the interaction between two LTP engines. LTP uses an underlying communication service as described in 5.2 to transmit and receive LTP segments.



**Figure 2-2: Communications View of LTP**

Figure 2-3 illustrates an LTP block transmission operation involving both red (reliable) and green (unreliable) parts. In the figure, the sender generates an asynchronous checkpoint (third red segment) to which the receiver responds with a report segment. The segment containing the EORP is lost, as well as the second green-part segment. The EORP segment is retransmitted; the lost green segment is not.



**Figure 2-3: Overview of LTP Interactions**

### 2.3.2 SERVICES NOT PROVIDED BY LTP

LTP does not provide an ‘in-order delivery’ service between different LTP blocks. That is, the order of arrival of LTP blocks may not be the order in which they were submitted to the sending LTP engine for transmission. This is especially true when multiple blocks are sent concurrently, as retransmissions of segments from the first block may cause the (red part of) that block to be delivered to the destination LTP client after a block that started transmission later.

#### NOTES

- 1 If in-order delivery is desired, the Delay-Tolerant Payload Conditioning (DTPC) service of the Bundle Protocol for CCSDS should be considered.
- 2 A sending LTP client application can ensure in-order delivery of a sequence of LTP blocks at the cost of performance by waiting for confirmation of the delivery of block  $N$  to the receiver (a `TransmissionSessionCompletion.indication`, see section 4.4.9) before submitting block  $N+1$  for transmission.

### 2.3.3 ADDRESSING

For CCSDS, every LTP engine deployed in space will have a unique engine ID. At each LTP engine location, address look-up capabilities are provided using information contained

in the associated MIB. This look-up capability provides translation between the engine ID and the information needed to communicate with that engine using the UCPs, which may in reality be an IP address, radio device buffer, APID, virtual channel number, or other implementation-specific mechanism.


## 3 CCSDS PROFILE OF RFC 5326

### 3.1 BASE SPECIFICATIONS

**3.1.1** This document adopts the Licklider Transmission Protocol (LTP) as specified in Internet RFC 5326 (reference [2]), with the constraints and exceptions specified in section 3 of this document.


**3.1.2** This document adopts the Licklider Transmission Protocol security extensions as specified in Internet RFC 5327 (reference [3]) with the constraints and exceptions specified in section 3 of this document.

### 3.2 AMBIGUITY RESOLUTION



**Ambiguities or contradictions** between the text of section 3 and the text of RFC 5326 or RFC5327 shall be resolved in favor of the RFC.

### 3.3 LTP OVER UDP



**3.3.1** Contrary to section 5 of RFC5326, **this document allows UDP to be used as an underlying communication service for LTP when deployed in private networks.**


NOTE – If LTP is deployed over UDP, then designers must consider the approach to be taken to control network congestion, since neither LTP nor UDP mandate mechanisms for congestion control. LTP implementations can provide rate-control, congestion control, or other mechanisms to mitigate the chance of congestion in shared networks. An alternative is to deploy LTP over the Datagram Congestion Control Protocol (DCCP) rather than directly over UDP.

**3.3.2** Implementations of LTP over UDP should use the ‘ltp-deepspace’ UDP port number, 1113 decimal, as specified in the Internet Assigned Numbers Authority (IANA) Port Number Registry (reference [4]).

NOTE – Individual sender/receiver pairs can choose alternate port numbers for communication. Using the ‘standard’ port of 1113 (ltp-deepspace) can facilitate the use of network management/debugging equipment and software that assumes LTP uses port 1113.



### 3.4 LTP FOR CCSDS



When used in support of CCSDS missions and across space links, **LTP should be deployed across individual space data links and should be terminated at the ends of each space data link.**

## NOTES

- 1 The LTP protocol was not designed to address issues associated with communication over a concatenation of multiple space data links with heterogeneous characteristics.
- 2 **When an underlying** communication service such as UDP that provides multi-hop data delivery, it may be desirable to extend LTP connections across multiple hops in the underlying network. This might especially be the case for LTP segments crossing the terrestrial Internet, for example.

### 3.5 LIMITS ON THE RANGES OF LTP FIELD VALUES

- 3.5.1** Session numbers chosen by sending LTP engines must be in the range  $[1, 2^{32}-1]$ .

## NOTES

- 1 It is suggested that CCSDS implementations choose sequential session numbers. The rationale for this suggestion is given in the notes below.
- 2 RFC 5326 section 3.1 states: “The format and resolution of session number are matters that are private to the LTP sending LTP engine; **the only requirement imposed by LTP is that every session initiated by an LTP engine must be uniquely identified by the session ID.**”
- 3 Green-part data is not reliably transmitted under LTP. In particular, **if there is green-part data in a block, the LTP segment containing the EOB marker may not be delivered to the destination.** As a consequence, if the session contains green-part data there is no way for a sending LTP engine to know when an LTP receiver has closed a session.
- 4 LTP does not require in-order delivery from the underlying system; thus it might be possible for LTP PDUs to be misordered by the underlying communication system. **If the sender re-uses a sessionID from a session that contained red-part data and then an old segment is delivered (either an old data segment to the receiver or an old report to the sender), it could cause errors.**
- 5 Thus in the general case where the underlying system may deliver PDUs out of order, **the only way within the bounds of the LTP protocol for the sender to ensure that it meets the requirements as stated in RFC 5326 is never to re-use LTP session identifiers.** It may be possible for mechanisms outside the protocol to determine when LTP session identifiers can be reused.
- 6 At a rate of 10 sessions per second continuously 24 hours per day, it would take roughly 13.6 years to consume the entire sessionID space. The Commercial Generic Bioprocessing Apparatus (CGBA) nodes on the international space station downlinked about a million files between 2005 and 2010, or roughly 25 per hour. At that rate the CGBA equipment could run for roughly 1,900 years before expending all  $2^{32}$  session IDs.

- 7 Ensuring uniqueness of session numbers across anomalies such as system restarts is outside the scope of this protocol.
- 8 Ensuring uniqueness of LTP session numbers across spacecraft anomalies such as system resets can be difficult. If there is persistent storage that is maintained across such anomalies, it can be used to store LTP session numbers used. If such storage is not available, other mechanisms will need to be invoked after anomalies to identify which LTP session numbers are available for future use. Such mechanisms might include time-based selection or management via mechanisms outside the LTP protocol.

**3.5.2** The initial checkpoint serial number values used by conformant implementations must be in the range  $[1, 2^{14}-1]$ .

**3.5.3** The initial checkpoint serial number values used by conformant implementations should be chosen at random.

**3.5.4** If the value of the checkpoint serial number for a given session exceeds  $2^{32}$ , the session sender must cancel the session with reason code SYS\_CNCLD.

**3.5.5** The initial report serial number values used by conformant implementations must be in the range  $[1, 2^{14}-1]$ .

**3.5.6** The initial report serial number values used by conformant implementations should be chosen at random.

**3.5.7** If the value of the report serial number for a given session exceeds  $2^{32}$ , the session receiver must cancel the session with reason code SYS\_CNCLD.

## NOTES

- 1 The above requirements (3.5.2, 3.5.4) make it easier to implement LTP on 32-bit processors and also impose a level of bit efficiency on CCSDS implementations of LTP. The resulting full ranges of allowable values for each of the fields can be encoded in SDNVs of at most five octets.
- 2 Limiting the initial report and serial number values to the range  $[1, 2^{14}-1]$  ensures that the initial values will fit into two-byte SDNVs.

## 3.6 AGENCY USE OF LTP ENGINE IDS

All instances of LTP for deployment in support of CCSDS missions must use LTP Engine IDs allocated by the CCSDS Space Assigned Numbers Authority (SANA) (reference [5]).

NOTE – LTP Engine IDs are represented in the LTP protocol with SDNVs. Values up to  $2^{14}-1$  can be represented in two bytes.

### **3.7 GREEN-PART DATA**



Support for green-part (unreliable) data is optional for CCSDS implementations. If an implementation supports any green-part operation (i.e., sending or receiving) then it must support both sending and receiving of green-part data.

### **3.8 LTP EXTENSIONS**

**3.8.1** Implementations must ignore LTP extensions that they do not know how to process on receipt.



**3.8.2** LTP implementations that transmit segments with LTP extensions must identify those extensions using the values prescribed by the IANA registry for LTP extension tags (reference [6]).

### **3.9 LTP SECURITY**

**3.9.1** Compliant LTP implementations may implement the authentication mechanisms defined in section 2.1 of RFC 5327.

**3.9.2** Compliant LTP implementations must not implement the cookie security extension defined in section 2.2 of RFC 5327.

**3.9.3** If authentication is implemented, elements of the MIB must dictate when particular security mechanisms must be used for sending, receiving, or both.

NOTE – The security policy that determines when particular security mechanisms must be invoked is outside the scope of this document.

**3.9.4** If an implementation provides LTP authentication service, it must identify the ciphersuite used in accordance with the IANA registry for LTP Ciphersuites (reference [6]).

**3.9.5** If LTP authentication is required for receiving data from a particular peer, the management information base must contain the key material to be used with that peer.



NOTE – The LTP authentication mechanism is designed to protect a receiver from a denial-of-service attack from malicious transmitters. Each sending LTP engine may use a single key to authenticate itself to all peers to which it transmits.

**3.9.6** If authentication is used, it must be included on either all LTP segments or none.



## **4 LTP SERVICE SPECIFICATION**

### **4.1 SERVICES AT THE USER INTERFACE**

**4.1.1** The following services provided by the protocol shall be made available to the LTP client:

- Initiate a data transfer;
- Cancel an ongoing data transfer;
- Receive transfer of data from a remote entity.

**4.1.2** Implementations may provide additional services beyond those described in section 4 of this document.

NOTE – Whether and how such services are invoked is an implementation matter. For example, an implementation might want to provide a service to deliver partial red-part data before the entire red-part of the block is received.



### **4.2 SUMMARY OF PRIMITIVES**

**4.2.1** The LTP service shall consume all the following request primitives:

- `Transmission.request;`
- `CancelTransmission.request;`
- `CancelReception.request.`

**4.2.2** The LTP service shall deliver the following indication primitives:

- `TransmissionSessionStart.indication;`
- `ReceptionSessionStart.indication;`
- `GreenPartSegmentArrival.indication;`
- `RedPartReception.indication;`
- `TransmissionSessionCompletion.indication;`
- `TransmissionSessionCancellation.indication;`
- `ReceptionSessionCancellation.indication;`
- `InitialTransmissionCompletion.indication.`

### **4.3 SUMMARY OF PARAMETERS**

#### **4.3.1 DESTINATION CLIENT SERVICE ID NUMBER**

The client service ID number identifies the layer-(N+1) service to which the segment is to be delivered by the receiving LTP engine that is providing the N-layer service.

#### **4.3.2 SOURCE LTP ENGINE ID**

The Source LTP engine ID is the LTP engine ID of the LTP engine that is the transmitter of data blocks.

#### **4.3.3 DESTINATION LTP ENGINE ID**

The Destination LTP engine ID is the LTP engine ID of the LTP engine that is to be the receiver of data blocks.

#### **4.3.4 CLIENT SERVICE DATA TO SEND**

Client Service Data to Send is the client data to be transmitted.

NOTE – It is assumed that this includes the client data itself as well as the length of the client data.

#### **4.3.5 SESSION ID**

**4.3.5.1** The session ID uniquely identifies, among all transmission sessions between the sender and receiver, the session to which the segment belongs.

**4.3.5.2** The session ID comprises the following:

- session originator: the engine ID of the sender;
- session number: as discussed in 3.5, above.

#### **4.3.6 REASON CODE**

Reason Code is an integer that identifies to a remote LTP engine the reason behind a particular action (typically the cancellation of a transmission).

NOTE – The LTP reason codes that may be carried in cancel segments are listed in section 3.2.4 of RFC 5326.

#### **4.3.7 OFFSET**

The offset of a byte within a block is the number of bytes that precede it in the block.

#### **4.3.8 LENGTH**

Length is the number of octets in a logical group of octets.

#### **4.3.9 GREEN PART BYTES / RED PART BYTES**

The Green Part (Red Part) data delivered by LTP to a destination LTP client.

NOTE – It is assumed that this includes the data itself as well as the length of the client data.

#### **4.3.10 END-OF-BLOCK INDICATIONS**

The RedPartReception and GreenPartReception indications include a notification to the receiver as to whether the received object is the end of an LTP block or not. Logically this is a Boolean value but is expressed in the service specification simply as an indication.

## 4.4 LTP SERVICE PRIMITIVES

### NOTES

- 1 Section 8.1 of RFC 5326 contains a state transition diagram for sending LTP engines.
- 2 Section 8.2 of RFC 5326 contains a state transition diagram for receiving LTP engines.

### 4.4.1 `Transmission.request`

#### 4.4.1.1 Function

The `Transmission.request` primitive shall be used by the LTP client to request delivery of a sequence of bytes to the destination client service.

#### 4.4.1.2 Semantics

4.4.1.2.1 `Transmission.request` shall provide parameters as follows:

```
Transmission.request( destination client service ID,  
                    destination LTP engine ID,  
                    client service data to send,  
                    length of the red-part of the data)
```

4.4.1.2.2 The value of length of the red-part of the data must be in the range from zero to the total length of data to be sent.

#### 4.4.1.3 When Generated

`Transmission.request` may be generated by the LTP client at any time.

#### 4.4.1.4 Effect on Receipt

4.4.1.4.1 Receipt of a `Transmission.request` shall cause the LTP engine to initiate transmission of the data.

#### 4.4.1.5 Discussion—Additional Comments

The `Transmission.request` results in the delivery of a `TransmissionSessionStart.indication` to the application so that the transmission may be subsequently uniquely identified.

The ability to send/receive green-part (unreliable) data is OPTIONAL in the CCSDS specification. If the implementation does not support green-part data then the length of the client service data to send must equal the length of the red-part of the data.

## **4.4.2 CancelTransmission.request**

### **4.4.2.1 Function**

The `CancelTransmission.request` primitive shall be issued by the LTP service client to request cancellation of transmission of a data block.

### **4.4.2.2 Semantics**

`CancelTransmission.request` shall provide parameters as follows:

`CancelTransmission.request (session ID)`

### **4.4.2.3 When Generated**

`CancelTransmission.request` may be generated by the LTP client at any time.

### **4.4.2.4 Effect on receipt**

The data transmission associated with the LTP session ID provided by the service client shall be stopped as described in section 4.2 of RFC 5326.

### **4.4.2.5 Discussion—Additional Comments**

None.

### **4.4.3 CancelReception.request**

#### **4.4.3.1 Function**

The `CancelReception.request` primitive shall be issued by the LTP service client to request cancellation of reception of a data block.

#### **4.4.3.2 Semantics**

`CancelReception.request` shall provide parameters as follows:

`CancelReception.request (session ID)`

#### **4.4.3.3 When Generated**

`CancelReception.request` may be generated by the LTP client at any time.

#### **4.4.3.4 Effect on receipt**

The data reception associated with the LTP session ID provided by the service client shall be stopped as described in section 4.2 of RFC 5326.

#### **4.4.3.5 Discussion—Additional Comments**

None.

#### **4.4.4 TransmissionSessionStart.indication**

##### **4.4.4.1 Function**

At the sender, the `TransmissionSessionStart.indication` primitive shall be used to inform the client service of the initiation of the transmission session.

##### **4.4.4.2 Semantics**

`TransmissionSessionStart.indication` shall provide parameters as follows:

`TransmissionSessionStart.indication (session ID)`

##### **4.4.4.3 When Generated**

At the sender, a `TransmissionSessionStart.indication` shall be generated by an LTP engine once the LTP engine has consumed a transmission request from the sender.

##### **4.4.4.4 Effect on Receipt**

The effect of this indication is dependent on the LTP client application that consumes it.

##### **4.4.4.5 Discussion—Additional Comments**

This indication is provided to the sending application so that the sending application can subsequently identify the transmission. Exactly how the indication is associated with a particular transmission request is an implementation matter.

On receiving this notice the client service may, for example, release resources of its own that are allocated to the block being transmitted, or remember the session ID so that the session can be canceled in the future if necessary.

#### **4.4.5 ReceptionSessionStart.indication**

##### **4.4.5.1 Function**

At the receiver, `ReceptionSessionStart.indication` primitive shall be used to indicate the beginning of a new reception session.

##### **4.4.5.2 Semantics**

`ReceptionSessionStart.indication` shall provide parameters as follows:

`ReceptionSessionStart.indication (session ID)`

##### **4.4.5.3 When Generated**

At the receiver, a `ReceptionSessionStart.indication` shall be generated by the LTP engine upon the arrival of the first data segment carrying a new session ID.

##### **4.4.5.4 Effect on Receipt**

The effect of this indication is dependent on the LTP client application that consumes it.

##### **4.4.5.5 Discussion—Additional Comments**

On receiving this notice the client service may, for example, remember the session ID so that the session can be canceled in the future if necessary.



## 4.4.6 GreenPartSegmentArrival.indication

### 4.4.6.1 Function

At the LTP receiver, a `GreenPartSegmentArrival.indication` primitive shall be used to indicate to the service client that a segment containing green (unreliable) data has been received and shall pass the received data to the service client.

### 4.4.6.2 Semantics

`GreenPartSegmentArrival.indication` shall provide parameters as follows:

```
GreenPartSegmentArrival.indication (session ID,  
                                     green-part bytes,  
                                     offset of the data segment's  
                                       content from the start of  
                                       the block,  
                                     indication as to whether or not  
                                       the last byte of this data  
                                       segment's content is also  
                                       the last byte of the block,  
                                     source LTP engine ID)
```

### 4.4.6.3 When Generated

A `GreenPartSegmentArrival.indication` shall be generated by the LTP engine on receipt of each green data segment.

### 4.4.6.4 Effect on Receipt

The effect of the receipt of a green segment is application-dependent.

### 4.4.6.5 Discussion—Additional Comments

Support for Green-Part data is optional in this specification.

## 4.4.7 RedPartReception.indication

### 4.4.7.1 Function

At the LTP receiver, a RedPartReception.indication primitive shall be used to indicate to the service client that the reception of the red-part of a block is complete and shall pass the data to the service client.

### 4.4.7.2 Semantics

RedPartReception.indication shall provide parameters as follows:

```
RedPartReception.indication ( session ID,  
                               red-part bytes,  
                               indication as to whether or not the  
                                 last byte of the red-part is also  
                                 the last byte of the block,  
                               source LTP engine ID)
```

### 4.4.7.3 When Generated

A RedPartReception.indication shall be generated by the LTP engine once the entire red-part of a block has been successfully received.

### 4.4.7.4 Effect on Receipt

The effect of receipt of a red-part is application-dependent.

### 4.4.7.5 Discussion—Additional Comments

None.

## **4.4.8 InitialTransmissionCompletion.indication**

### **4.4.8.1 Function**

An `InitialTransmissionCompletion.indication` primitive shall be used to inform the sending client service that all segments of a block (both red-part and green-part) have been transmitted.

### **4.4.8.2 Semantics**

`InitialTransmissionCompletion.indication` shall provide parameters as follows:

`InitialTransmissionCompletion.indication` (session ID)

### **4.4.8.3 When Generated**

`InitialTransmissionCompletion.indication` shall be generated by an LTP engine once all segments of a block have been transmitted.

### **4.4.8.4 Effect on Receipt**

The effect of `InitialTransmissionCompletion.indication` is service-client-dependent.

### **4.4.8.5 Discussion—Additional Comments**

This notice indicates only that original transmission is complete; retransmission of any lost red-part data segments may still be necessary.

## 4.4.9 TransmissionSessionCompletion.indication

### 4.4.9.1 Function

A `TransmissionSessionCompletion.indication` primitive shall be used to inform the client service that all bytes of the indicated data block have been transmitted *and* that the receiver has received the red-part of the block.

### 4.4.9.2 Semantics

`TransmissionSessionCompletion.indication` shall provide parameters as follows:

`TransmissionSessionCompletion.indication` (session ID)

### 4.4.9.3 When Generated

`TransmissionSessionCompletion.indication` shall be generated by a sending LTP engine once all bytes of the session have been transmitted and the receiver has indicated that all bytes of the red-part of the block have been received correctly.

### 4.4.9.4 Effect on Receipt

The application may, for example, choose to release resources associated with the transmission once all of the bytes have been successfully transmitted.

NOTE – This does not mean that all bytes have been received correctly at the destination.

### 4.4.9.5 Discussion—Additional Comments

(See also `InitialTransmissionCompletion.indication`.)

## **4.4.10 TransmissionSessionCancellation.indication**

### **4.4.10.1 Function**

A `TransmissionSessionCancellation.indication` primitive shall be used to inform the sending client service that the indicated session was terminated.

### **4.4.10.2 Semantics**

`TransmissionSessionCancellation.indication` shall provide parameters as follows:

```
TransmissionSessionCancellation.indication (    session ID,  
                                              reason code)
```

### **4.4.10.3 When Generated**

`TransmissionSessionCancellation.indication` shall be generated by a sending LTP engine when a session is terminated either by the receiver or as the result of an error or a lack of resources in the local LTP engine.

### **4.4.10.4 Effect on Receipt**

The effect of the `TransmissionSessionCancellation.indication` is application-dependent. An application might record the cancellation, declare an error state, and/or attempt to retransmit the data.

### **4.4.10.5 Discussion—Additional Comments**

When a sender receives a `TransmissionSessionCancellation.indication` there is no assurance that the destination client service instance received any portion of the data block.

#### **4.4.11 ReceptionSessionCancellation.indication**

##### **4.4.11.1 Function**

A `ReceptionSessionCancellation.indication` primitive shall be used to inform the receiving client service that the indicated session was terminated.

##### **4.4.11.2 Semantics**

`ReceptionSessionCancellation.indication` shall provide parameters as follows:

```
ReceptionSessionCancellation.indication ( session ID,  
                                         reason code)
```

##### **4.4.11.3 When Generated**

`ReceptionSessionCancellation.indication` shall be generated by a receiving LTP engine when a session is canceled either by the sender or as the result of an error or a resource quench condition in the local LTP engine.

##### **4.4.11.4 Effect on Receipt**

The effect of `ReceptionSessionCancellation.indication` is service-client-dependent.

##### **4.4.11.5 Discussion—Additional Comments**


No subsequent indications will be issued for this session.

## 5 SERVICES LTP REQUIRES OF THE SYSTEM

### 5.1 RELIABLE STORAGE SERVICE


LTP engines require access to a reliable storage service.

#### NOTES

- 
- 1 Reliable storage is required even if unreliable (green-part) data is sent, since LTP may need to store the data between contacts with a peer (i.e., while timers are suspended).
  - 2 This storage mechanism may be in dynamic memory or via a persistent mechanism such as a solid-state recorder or file system.
  - 3 The implementation of this storage may be shared among multiple elements of the communication stack so that reliability mechanisms at multiple layers do not have to maintain multiple copies of the data being transmitted.
  - 4 Volume of storage required and duration of storage are mission and implementation dependent.

### 5.2 UNDERLYING COMMUNICATION SERVICE REQUIREMENTS

**5.2.1** This document adopts the functional requirements on the underlying communication layers as described in section 5 of RFC 5326, namely:



**5.2.2** The LTP authentication extension should be used when the likelihood of datagram corruption by the underlying communication service is non-negligible.

NOTE – The LTP authentication mechanism provides an integrity service as a by-product.

**5.2.3** LTP assumes that the service provided by the protocols underlying LTP (not necessarily only the layer-(N-1) protocol) provides a service that is:


- without error (that is, all bits of any layer-(N-1) service data unit that arrives at a destination LTP engine are guaranteed to have the values that were originally transmitted by the source LTP engine);
- possibly incomplete (that is, not all transmitted layer-(N-1) service data units are guaranteed to arrive);
- possibly ‘not in sequence’ (that is, the order in which layer-(N-1) service data units arrive is not guaranteed to be the order in which they were transmitted).

**5.2.4** **The following information must be available to LTP**, either from the local operating environment or from the underlying communication service provider:



## NOTES


- 1 The means by which this information is accessed by LTP is implementation-dependent.
- 2 These capabilities may be provided via a MIB:
  - indications of when the link to a specific LTP destination transitions between being available and being unavailable for LTP transmissions;
  - schedule of times when a remote LTP engine is set to commence or terminate communication with the local engine based on the engines' operating schedules;
  - for each transmission opportunity to a peer, the maximum LTP segment size that should be used to send LTP segments during that opportunity;
  - for each transmission opportunity to a peer, the nominal data rate available to LTP for transmission;
  - for each reception opportunity from a peer, the nominal data rate at which LTP data will be received;
  - the current distance (in light seconds) to any peer engine in order to calculate timeout intervals;
  - the times when the underlying communication service provider transmits a checkpoint, report segment, or cancel segment so that timers can be started.

 **5.2.5** The layer-(N-1) service must deliver only **complete layer-(N-1) SDUs** (segments) to the receiving LTP engine.

**5.2.6** The requirement in RFC 5326 that each layer-(N-1) SDU must carry an integral number of LTP segments in each layer-(N-1) PDU is not included here.

NOTE – **So long as the layer-(N-1) service provides only complete layer-N (LTP) PDUs to the receiving LTP engine, LTP does not care how the layer N-1 service is implemented.**

**5.2.7** If the service presented to LTP by layer-(N-1) provides integrity checking, then it must discard layer-(N-1) SDUs that are determined to be corrupted at the receiver.

 NOTE – LTP assumes that, if necessary to the overall design, the service presented to LTP by layer-(N-1) provides a cap on the rate at which a sending LTP engine can inject data into the layer N-1 service. If such a capability is needed and is not provided by the layer N-1 service, it may be possible to provide it as part of the LTP interface to the layer N-1 service.



## 6 CONFORMANCE REQUIREMENTS

### 6.1 PICS PROFORMA


An implementer shall prepare a Protocol Implementation Conformance Statement (PICS) based on the defined proforma in annex C of this document.

### 6.2 LICKLIDER TRANSMISSION PROTOCOL REQUIREMENTS

#### 6.2.1 MAJOR CAPABILITIES


##### 6.2.1.1 All LTP Engines

All LTP engines must implement the following capabilities in accordance with the base standard:

- 
- a) LTP segment structure as described in RFC 5326 sections 3.0, 3.1, 3.1.4, 3.2.1, 3.2.2, 3.2.3, 3.2.4, 3.3, 4.1, 4.2, 6.0, 6.1, 6.2, 6.3, 6.4, 6.5, 6.5, 6.6, 6.7, 6.8, 6.9, 6.11, 6.12, 6.13, 6.14, 6.15, 6.16, 6.17, 6.18, 6.19, 6.20, 6.21, 7.1, 7.3, 7.4, 7.5, 7.6, 7.7, 8.1 [as pertains to red data], 8.2 [as pertains to red data], 10.2;
  - b) network management information described in annex B of this document.

NOTE – Section 2.0 of RFC 5326 is included above specifically to cover the definition of SDNVs. Self-Delimiting Numeric Values are also defined in RFC 6256 (reference [E5]).

##### 6.2.1.2 LTP Authentication



LTP implementations may implement the authentication measures described in 3.1.2, 3.9.1, and 3.9.3—3.9.6 of this document and section 2.1 of RFC 5327 (reference [3]).

##### 6.2.1.3 LTP Security Cookies

LTP implementations must NOT implement the LTP cookie mechanism described in section 2.2 of RFC5327.

##### 6.2.1.4 LTP over UDP

Implementations that run using UDP as an underlying communication service must do so according to 3.3.1 and 3.3.2 of this document, and section 10.1 of RFC5326.

#### **6.2.1.5 Session Number Selection**

LTP session numbers must be chosen according to 3.5.1 of this document.

#### **6.2.1.6 Checkpoint Serial Number Selection**

The checkpoint serial numbers must be chosen according to 3.5.2 and 3.5.3 of this document.

#### **6.2.1.7 LTP Extensions**

Implementations must ignore unknown extensions on receipt as described in 3.8.1 of this document.

#### **6.2.1.8 LTP Encapsulation in CCSDS Encapsulation Packets**

Implementations that use CCSDS Encapsulation Packets as an underlying communication mechanism must do so as described in A2 of this document.

#### **6.2.1.9 LTP Encapsulation in Space packets**

Implementations that use CCSDS Space Packets as an underlying communication mechanism must do so as described in A3 of this document.

#### **6.2.1.10 Report Serial Number Selection**

The checkpoint serial numbers must be chosen according to 3.5.5 and 3.5.6 of this document.

#### **6.2.1.11 Green (Unreliable) Data**

If an implementation supports sending or receiving of green (unreliable) data, it must support both sending and receiving of green data as described in sections 6.1, 6.10, and 7.2 of RFC5326, and the portions of sections 8.1 and 8.2 of RFC5326 that pertain to green data.

#### **6.2.1.12 LTP Service Data Aggregation**

LTP implementations must implement Service Data Aggregation in accordance with the LTP Service Data Aggregation Client Operations described in this document. In particular

- a) the client service interface of 7.2.1 of this document;
- b) service data formatted according to 7.2.2.2 of this document;
- c) the procedures described in 7.2.2.3 through 7.2.2.5.2, inclusive, of this document.

## **6.2.2 NETWORK MANAGEMENT REQUIREMENTS**

Conformant systems shall maintain the management information identified in annex B of this document. Neither the method for remote access to this information (e.g., via the Simple Network Management Protocol [SNMP]) nor the format of this information for remote transmission (e.g., ASN.1 Basic Encoding Rules) are specified by this document.

## 7 CLIENT OPERATIONS

### 7.1 OVERVIEW—LTP SERVICE DATA AGGREGATION (SDA)

This section describes standard interoperable procedures which are not included in LTP itself but which use the services provided by LTP.

Since LTP acknowledgements are issued, at minimum, at ‘block’ granularity, the volume of positive and negative acknowledgment traffic generated by LTP may be roughly regulated by controlling the minimum size of the ‘red parts’ of the blocks transmitted. Because retransmission is performed at ‘segment’ granularity, the amount of retransmitted data sent in response to corrupted/lost segments can be roughly controlled by adjusting the size of the LTP segments.

However, by definition each LTP block contains exactly one client service data unit. Requiring a lower limit on the size of LTP client SDUs in order to increase block size and thereby minimize acknowledgement traffic would be unreasonable.

Instead, this section defines a standard *LTP Service Data Aggregation (SDA)* client service to which LTP clients may present client data units of arbitrary size, which will aggregate small client data units (as necessary) into service data units whose size is normally no less than some asserted minimum LTP block red-part size.

That is, LTP clients may interact with SDA rather than with LTP itself. SDA will aggregate client data units into service data units as necessary, present those possibly aggregated service data units to LTP for transmission, acquire possibly aggregated service data units received by LTP, extract individual client data units from those service data units as necessary, and present the received client data units to the client.

Only client data units requiring assured transmission may be presented to SDA; that is, the ‘red length’ of each client data unit presented to the SDA must be equal to the total length of that client data unit.

### 7.2 LTP SDA SPECIFICATION

#### 7.2.1 SERVICE INTERFACE

**7.2.1.1** LTP SDA shall present to clients the `Transmission.request`, `InitialTransmissionCompletion.indication` and `RedPartReception.indication` primitives that are defined in the LTP service specification.

**7.2.1.2** Only `Transmission.request` requests where the length of the red-part of the data is equal to the total length of the data to be transmitted will be accepted by the SDA service. Actions taken by the service for other requests is an implementation matter.

NOTE – Any data loss such as would be possible if part of the SDA PDU were sent as green data might make it impossible to recover all client data units at the receiver. A receiver attempting to cache and reorder green data segments to reconstruct the client data units they contain might also be vulnerable to a denial-of-service attack where multiple green data segments are received but not enough information is provided to extract any client data units.

## **7.2.2 PROCEDURES**

### **7.2.2.1 Client Service ID**

The client service ID passed by SDA to LTP shall be '2', signifying 'LTP Service Data Aggregation'.

### **7.2.2.2 Client Service Data**

**7.2.2.2.1** The client service data unit passed by SDA to LTP shall be a single aggregated SDA service data unit. Each aggregated SDA service data unit shall be the concatenation of one or more SDA **client data capsules**. Each SDA client data capsule shall comprise:

- a single SDNV containing the LTP client service ID passed to SDA by the client in a `Transmission.request` primitive, e.g., '1' signifying 'Bundle Protocol';
- a single complete client data unit as passed to SDA by the client in that same `Transmission.request` primitive.

NOTE – The different SDA capsules contained in an aggregated SDA service unit may have different client service identifiers.

**7.2.2.2.2** The entire client service data unit passed by SDA to LTP (an aggregated SDA service data unit) shall be 'red' (reliably transmitted) data.

NOTE – The length of a client data unit in an SDA service data unit is not constrained in any way by the lengths of the LTP data *segments* in which portions of the service data will be transmitted.

**7.2.2.2.3** A sending LTP engine may impose an upper limit on the red length of an SDA service data unit. Mechanisms by which SDA may determine the red length limit are an implementation matter.


### **7.2.2.3 Session Start**

Procedures to be performed by SDA upon reception of a `TransmissionSessionStart.indication/ReceptionSessionStart.indication` from the LTP engine are an implementation matter.

## **7.2.2.4 Transmission**

### **7.2.2.4.1 Transmission Initiation**

#### **7.2.2.4.1.1 General**



Upon reception of a `Transmission.request` primitive from the client, SDA shall retain the client service ID and the client data unit in an SDA client data capsule for inclusion in an aggregated SDA service data unit destined for the indicated remote LTP engine.

#### **7.2.2.4.1.2 Size Limit Reached**

**7.2.2.4.1.2.1** If the sum of the lengths of all SDA client data capsules currently retained for inclusion in the next aggregated SDA service data unit for the indicated remote LTP engine is now greater than or equal to the configured service data unit size threshold for transmission to that engine, then SDA shall submit a `Transmission.request` primitive to the LTP engine.

**7.2.2.4.1.2.2** The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.

**7.2.2.4.1.2.3** The `Transmission.request` presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.

**7.2.2.4.1.2.4** The SDA client data capsules included in the service data unit for the `Transmission.request` to LTP shall be removed from the SDA set of retained client data.

#### **7.2.2.4.1.3 Time Limit Reached**

**7.2.2.4.1.3.1** When the difference between the current time and the earliest time at which an SDA client data capsule was retained for inclusion in the next aggregated SDA service data unit for some remote LTP engine exceeds the configured service data aggregation interval threshold for transmission to that engine, SDA shall submit a `Transmission.request` primitive to the LTP engine.

**7.2.2.4.1.3.2** The service data unit for this primitive shall be the aggregated SDA service data unit for the indicated remote LTP engine as described above.

**7.2.2.4.1.3.3** The `Transmission.request` presented by SDA to LTP shall indicate that the entire SDA PDU is comprised of 'red' data.

**7.2.2.4.1.3.4** The SDA client data capsules included in the `Transmission.request` to LTP shall be removed from the SDA set of retained client data.

#### **7.2.2.4.2 Completion of Initial Transmission**

Procedures to be performed by SDA upon reception of an InitialTransmissionCompletion.indication primitive are an implementation matter.

#### **7.2.2.4.3 Transmission Completion**

Upon reception of a TransmissionSessionCompletion.indication primitive from the LTP engine, SDA shall issue a TransmissionSessionCompletion.indication primitive for each SDA client data capsule in the SDA service data unit.

#### **7.2.2.4.4 Transmission Cancellation**

Procedures to be performed by SDA upon reception of a TransmissionSessionCancellation.indication primitive from the LTP receiver engine are an implementation matter, but in particular SDA shall not issue a TransmissionSessionCompletion.indication primitive for any SDA client data capsule in the SDA service data unit.

#### **7.2.2.5 Reception**

##### **7.2.2.5.1 Red-Part Reception**

Upon reception of a RedPartReception.indication from the LTP engine, SDA shall extract all SDA client data capsules from the service data and deliver the encapsulated client data units to the indicated clients in RedPartReception.indication primitives. The manner in which SDA client data capsules are extracted from the service data shall be dependent upon the client service ID noted at the start of each capsule but is otherwise an implementation matter.

##### **7.2.2.5.2 Reception Cancellation**

Upon reception of a ReceptionSessionCancellation.indication primitive from the sending LTP engine, SDA shall deliver one ReceptionSessionCancellation.indication for each SDA client data capsule in the service data of the canceled LTP transaction; each such indication shall be delivered to the client identified by the client data capsule's client service ID.

## ANNEX A

### USING THE CCSDS SPACE PACKET OR ENCAPSULATION SERVICE AS AN UNDERLYING COMMUNICATION SERVICE FOR LTP

#### (NORMATIVE)

#### A1 OVERVIEW

Ensuring interoperability between two instances of LTP operating over a particular underlying communication service requires knowledge of how LTP segments are inserted into and extracted from the underlying service's PDUs. This annex specifies how LTP segments are to be carried over the CCSDS Encapsulation Service (reference [7]) and the CCSDS Space Packet Service (reference [9]).

This annex does not define any new protocol mechanisms; it specifies the way in which an LTP implementation **must invoke the existing capabilities of the Encapsulation Service and Space Packet Service.**

#### A2 CARRYING LTP SEGMENTS USING THE CCSDS ENCAPSULATION SERVICE

**A2.1** When the CCSDS Encapsulation Service (reference [7]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Encapsulation Service.

**A2.2** The protocol identifier for the Encapsulation Service to be used to identify Encapsulation Packets carrying LTP segments as their payloads should be that specified in the SANA Protocol Identifier for Encapsulation Service registry (reference [8]).

NOTE – The CCSDS Encapsulation Service defines mechanisms for encapsulating SDUs in CCSDS Space Packets as well as CCSDS Encapsulation Packets. **Which underlying (below encapsulation) service is used will affect the parameters used to invoke the encapsulation.request function of the Encapsulation Service.** In particular, if CCSDS Space Packets are used beneath encapsulation (Packet Version Number [PVN]=1), the Encapsulation Protocol Identifier (EPI) parameter of the encapsulation.request invocation will be an APID; if CCSDS Encapsulation Packets are used (PVN=8), the EPI parameter will be a protocol ID identifying LTP.



### **A3 CARRYING LTP SEGMENTS USING THE CCSDS SPACE PACKET SERVICE**

**A3.1** When the Space Packet Service (reference [9]) is used as the underlying communication service to transport LTP segments, one LTP segment shall be the SDU of the Space Packet Service.

**A3.2** The APID to be used to identify Space Packets carrying LTP segments as their payloads should be that specified in the SANA Space Packet Protocol Application Process Identifier (APID) registry (reference [10])

## **ANNEX B**

### **LICKLIDER TRANSMISSION PROTOCOL MANAGEMENT INFORMATION BASE**

#### **(NORMATIVE)**

#### **B1 BASIC REQUIREMENTS**

The operation of each LTP engine shall be supported by a single Management Information Base (MIB) comprising the items of information described below.

#### NOTES

- 1 The MIB described here is not defined in RFC 5326.
- 2 Representation of, and mechanisms for access to, MIB items will be implementation matters. In particular, determination of which items will be static and which will be dynamic is a matter of implementation.

#### **B2 LOCAL ENGINE CONFIGURATION INFORMATION**

For each item of local engine configuration information (see table B-1), a single value shall apply and shall pertain to the entire LTP engine.

**Table B-1: Local Engine Configuration Information**

Item	Comment
Local Engine ID	The LTP engine ID of the local (i.e., this) LTP engine.
Checkpoint retransmission limit	As described in section 6.7 of RFC 5326.
Report segment retransmission limit	As described in section 6.8 of RFC 5326.
Reception problem limit	As described in section 6.11 of RFC 5326.
Cancellation segment retransmission limit	As described in section 6.16 of RFC 5326.
Retransmission cycle limit	As described in section 6.22 of RFC 5326.
Local queuing and processing delay	For use in computing timer intervals; allowance for queuing and processing delay at local engine.
Local operating schedule	Schedule of times that the local LTP engine expects to be operating (able to communicate with remote engines).
SDA Aggregation Size	The maximum amount of data (bytes) that will be aggregated by the SDA service before an LTP block is transmitted (7.2.2.4.1.2).
SDA Aggregation Time	The amount of time that service data units will be aggregated by the SDA service before an LTP block is transmitted (7.2.2.4.1.3).
Implements green part data	True if this LTP engine supports transmission / reception of green-part (unreliable) data, otherwise False.

### **B3 REMOTE ENGINE CONFIGURATION INFORMATION**

For each item of remote engine configuration information (see table B-2), a single value shall apply for each remote engine with which the local LTP engine may communicate.

**Table B-2: Remote Engine Configuration Information**

Item	Comment
Remote engine ID	The remote LTP engine ID.
UCP address	UCP Address to use when transmitting to this engine during this contact.
Maximum segment length	The maximum segment length that the remote implementation supports, in octets.
One-way light time (outbound)	For use in computing timer intervals; one-way light time TO the remote engine from the local one.
One-way light time (inbound)	For use in computing timer intervals; one-way light time FROM the remote engine to the local one.
Remote queuing and processing delay	For use in computing timer intervals; allowance for queuing and processing delay at remote engine.
Remote operating schedule	Schedule indicating when the remote engine is expected to be communicating with the local one. This information may or may not be provided by the MIB but is listed here for clarity; there may be other ways for the engine to determine the remote operating schedule.
Security: use authentication when sending	This determines if the local LTP engine uses the LTP Security authentication mechanism when communicating with this remote entity.
Security: allowable authentication ciphersuites	Identifies the authentication ciphersuites the local engine can use when communicating with the remote engine.
Security: sending authentication keys	For each ciphersuite the local LTP engine uses when communicating with the given remote engine, this identifies the authentication material (key) that should be used. NOTE – Implementations may wish to use a single key per ciphersuite for all peers.
Security: require authentication on incoming sessions	If true, then at least the initial segments received in sessions from this remote engine must carry LTP authentication extensions.
Security: receiving authentication keys	For each ciphersuite that may be used to communicate from the remote LTP node to the local one, this identifies the authentication material that should be used to authenticate incoming LTP segments containing LTP authentication extensions.

NOTES

- 1 Different one-way light times for outbound and inbound communications are included more to allow for differences in communications mechanisms in the two directions than for differences in the actual light times.
- 2 This assumes that remote engines use a single underlying communications protocol (so that the segment length is a function of the remote engine and does not depend on which underlying communications protocol is used).

## ANNEX C

### **PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT PROFORMA**

#### **(NORMATIVE)**

#### **C1 OVERVIEW**

This annex provides the Protocol Implementation Conformance Statement (PICS) Requirements List (RL) for CCSDS-compliant implementations of LTP. The PICS for an implementation is generated by completing the RL in accordance with the instructions below. An implementation shall satisfy the mandatory conformance requirements of the base standards referenced in the RL.

An implementation's completed RL is called the PICS. The PICS states which capabilities and options of the protocol have been implemented. The following can use the PICS:

- a) the protocol implementer, as a checklist to reduce the risk of failure to conform to the standard through oversight;
- b) the supplier and acquirer or potential acquirer of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis for understanding provided by the standard PICS proforma;
- c) the user or potential user of the implementation, as a basis for initially checking the possibility of interworking with another implementation (it should be noted that, while interworking can never be guaranteed, failure to interwork can often be predicted from incompatible PICSes);
- d) a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation.

#### **C2 INSTRUCTIONS FOR COMPLETING THE RL**

An implementer shows the extent of compliance to the protocol by completing the RL; that is, compliance to all mandatory requirements and the options that are not supported are shown. The resulting completed RL is called a PICS. In the Support column, each response shall be selected either from the indicated set of responses, or it shall comprise one or more parameter values as requested. If a conditional requirement is inapplicable, N/A should be used. If a mandatory requirement is not satisfied, exception information must be supplied by entering a reference Xi, where i is a unique identifier, to an accompanying rationale for the noncompliance.

### C3 NOTATION

**C3.1** The following symbols are used in the RL to indicate the status of features.

**Table C-1: PICS Notation**

Symbol	Meaning
M	Mandatory
M.<n>	Support of every item of the group labeled by the same number <n> required, but only one is active at a time
O	Optional
O.<n>	Optional, but support of at least one of the group of options labeled by the same numeral <n> is required
C	Conditional
--	Non-applicable field/function (i.e., logically impossible in the scope of the RL)
I	Out of scope of the RL (left as an implementation choice)
X	Excluded or prohibited

**C3.2** Two-character combinations may be used for dynamic conformance requirements. In this case, the first character refers to the static (implementation) status, and the second refers to the dynamic (use) status; thus ‘MO’ means ‘mandatory to be implemented, optional to be used’.

**C3.3** The following notations for conditional status shall be used.

**Table C-2: PICS Conditional Status Notation**

Symbol	Meaning
<predicate>: :	This notation introduces a group of items, all of which are conditional on <predicate>.
<predicate>:	This notation introduces a single item which is conditional on <predicate>.
<index>:	This notation indicates that the status following it applies only when the PICS states that the features identified by the index are supported. In the simplest case, <index> is the identifying tag of a single RL item. The symbol <index> also may be a Boolean expression composed of several indices.
<index>::	This notation indicates that the associated clause should be completed.

**C3.3.1** Either of the predicate forms may identify a protocol feature, or a Boolean combination of predicates.

**C3.3.2** ‘^’ is the symbol for logical negation, ‘|’ is the symbol for logical OR, and ‘&’ is the symbol for logical AND.

**C3.4** The following notations shall be used in the ‘Protocol Feature’ column.

**Table C-3: Symbols for PICS ‘Protocol Feature’ Column**

Symbol	Meaning
<r>	Denotes the receiving system.
<t>	Denotes the transmitting system.

**C3.5** The following symbols shall be used in the ‘Support’ column of the PICS.

**Table C-4: Symbols for PICS ‘Support’ Column**

Symbol	Meaning
Y	Yes, the feature is supported by the implementation.
N	No, the feature is not supported by the implementation.
N/A	The item is not applicable.

#### **C4 REFERENCED BASE STANDARDS**

**C4.1** The base standards referenced in the RL shall be:

- a) CCSDS LTP (this document);
- b) RFC 5326 (reference [2]).

**C4.2** In the tables below, the notation in the Reference column combines one of the short-form document identifiers above (e.g., CCSDS-LTP) with applicable subsection numbers in the referenced document. RFC numbers are used to facilitate reference to subsections within the Internet specifications.

#### **C5 GENERAL INFORMATION**

##### **C5.1 IDENTIFICATION OF PICS**

<b>Ref</b>	<b>Question</b>	<b>Response</b>
1	Date of Statement (DD/MM/YYYY)	
2	PICS serial number	
3	System conformance statement cross-reference	



**C5.2 IDENTIFICATION OF IMPLEMENTATION UNDER TEST (IUT)**

Ref	Question	Response
1	Implementation name	
2	Implementation version	
3	Name of hardware (machine) used in test	
4	Version of hardware (machine) used in test	
5	Name of operating system used during test	
6	Version of operating system used during test	
7	Additional configuration information pertinent to the test	
8	Other information	

**C5.3 IDENTIFICATION**

Ref	Question	Response
1	Supplier	
2	Point of contact for queries	
3	Implementation name(s) and version(s)	
4	Other information necessary for full identification (e.g., name(s) and version(s) for machines and/or operating systems	

**C5.4 PROTOCOL SUMMARY**

Ref	Question	Response
1	Protocol version	
2	Addenda implemented	
3	Amendments implemented	
4	Have any exceptions been required? NOTE – A YES answer means that the implementation does not conform to the protocol. Non-supported mandatory capabilities are to be identified in the PICS, with an explanation of why the implementation is non-conforming.	1. Yes  2. No
4	Date of statement (DD/MM/YYYY)	

**C6 REQUIREMENTS**

Item	Protocol Feature	Reference	Status	Support
baseLTP	The mechanisms required to effect Red-Part data transmission/reception from RFC 5326 except as noted in section 3 of this document.	This document, 3.1.1, 3.2–3.8.  RFC5326 sections 3.0, 3.1, 3.1.4, 3.2.1, 3.2.2, 3.2.3, 3.2.4, 3.3, 4.1, 4.2, 6.0, 6.1, 6.2, 6.3, 6.4, 6.5, 6.5, 6.6, 6.7, 6.8, 6.9, 6.11, 6.12, 6.13, 6.14, 6.15, 6.16, 6.17, 6.18, 6.19, 6.20, 7.1, 7.3, 7.4, 7.5, 7.6, 7.7, 8.1 [as pertains to red data], 8.2 [as pertains to red data], 10.2	M	

<b>Item</b>	<b>Protocol Feature</b>	<b>Reference</b>	<b>Status</b>	<b>Support</b>
LTPSec	LTP Security Extensions from RFC 5327 except as noted in section 3 of this document.	This document, 3.1.2, 3.9. RFC5327 section 2.1	O	
ltpUDP	LTP over User Datagram Protocol.	3.3 of this document. RFC5326 section 10.1	O	
sessionNo	LTP Session Number selection.	3.5.1 of this document.	M	
initCheckpointSerNo	Initial Checkpoint Serial Number selection.	3.5.2–3.5.3 of this document.	M	
checkpointSerNo	Cancel sessions when checkpoint serial number exceeds $2^{32}$ .	3.5.4 of this document.	M	
initReportSerNo	Initial Report Serial Number selection.	3.5.5–3.5.6 of this document.	M	
reportSerNo	Cancel sessions when report serial number exceeds $2^{32}$ .	3.5.7 of this document.	M	
ltpUnknownExt	Ignore unknown extensions on receipt.	3.8.1 of this document.	M	
ltpExt	LTP extensions in outbound segments identified correctly.	3.8.2 of this document.	M	
ltpCookie	Implementations must <u>NOT</u> use the LTP Cookie mechanism described in section 2.2 of RFC5327.	3.9.2 of this document.	M	
authMIB	Authentication information in managed information base.	3.9.3 and 3.9.4 of this document.	< ltpAuth >:M	

**CESG APPROVAL COPY - NOT FOR DISTRIBUTION**  
**CCSDS RECOMMENDED STANDARD FOR LICKLIDER TRANSMISSION PROTOCOL**

<b>Item</b>	<b>Protocol Feature</b>	<b>Reference</b>	<b>Status</b>	<b>Support</b>
ltpEncap	LTP encapsulation in CCSDS Encapsulation Packets.	A2 of this document.	O	
ltpSpacePacket	LTP encapsulation in CCSDS Space Packets.	A3 of this document	O	
supportsGreen	The implementation supports transmission and reception of green-part (unreliable) data	This document, 3.7 RFC 5326 section 6.1 and 8.1 [as pertain to green data.] RFC 5326 sections 6.10, 7.2 and 8.2 [as pertain to green data].	O	
serviceDataAggregation	SDA presents the Transmission.request, InitialTransmissionCompletion.indication, and RedPartReception.indication primitives that are part of the LTP specification	7.2 of this document	M	


## **ANNEX D**

### **SECURITY, SANA, AND PATENT CONSIDERATIONS**

#### **(INFORMATIVE)**

#### **D1 SECURITY CONSIDERATIONS**

##### **D1.1 OVERVIEW**



This document adopts the LTP Security Extensions RFC whereby a receiver can verify that a particular LTP segment was in fact sent by a known sender. This authentication mechanism can be used to protect against false or malicious data being sent to an LTP receiver. LTP as specified in this document does not provide any mechanisms for confidentiality, integrity, or non-repudiation of data. It is assumed that these services, if required, are provided by other layers of the protocol stack. The Bundle Security Protocol, for example, can be used above LTP and defines mechanisms to provide confidentiality and integrity services between security endpoints, which may be the ultimate source and destination(s).

NOTE – The LTP Security Extensions RFC defines an authentication mechanism that is adopted here and a ‘cookie’ mechanism to defend against denial-of-service attacks. The cookie mechanism does not add significant protection against denial-of-service attacks when LTP is deployed over individual links, as is recommended in this document, and adds considerable complexity to implementations. The LTP Security Extensions RFC does not define any mechanisms for data confidentiality, integrity (except as a by-product of authentication), or non-repudiation.

##### **D1.2 SECURITY CONCERNS WITH RESPECT TO THE CCSDS DOCUMENT**

###### **D1.2.1 Data Privacy**

This specification does not provide any mechanism to ensure data privacy. Any such mechanisms, if they are needed by missions, must be applied at other layers of the stack.

NOTE – The Bundle Security Protocol, part of the Delay / Disruption Tolerant Networking (DTN) protocol suite, defines mechanisms to provide confidentiality between ‘security endpoints’, which may be the ultimate source and destination(s) of data transmission.

### **D1.2.2 Data Integrity**

LTP security extensions do not explicitly provide a mechanism for insuring integrity.

NOTE – If authentication is used and is present for a particular LTP segment, it will provide integrity checking, since corrupted segments will fail to authenticate.

### **D1.2.3 Authentication of Communicating Entities**

The CCSDS profile of LTP defined in this document allows the use of the LTP authentication mechanisms defined in reference [3].

NOTE – The authentication mechanism allows a receiver to authenticate the sender(s) of individual LTP segments. This authentication incurs some additional overhead which varies depending on the authentication mechanism used.

### **D1.2.4 Control of Access to Resources**

This Recommended Standard assumes that control of access to resources will be managed by the systems executing the protocol. No provisions are made by the protocol described in this document to limit or control access to resources (e.g., CPU, storage, or bandwidth) used by the protocol.

### **D1.2.5 Availability of Resources**

If sufficient resources are not available for LTP to carry out a data transfer, the local LTP client service is notified via the service interface and mechanisms defined in the protocol (cancellation segments) are used to attempt to inform the remote LTP engine and client, and to cancel the data transfer.

### **D1.2.6 Auditing of Resource Usage**

No mechanisms are defined in this specification to audit or assist with the auditing of resource usage by the protocol.

### **D1.2.7 Potential Threats and Attack Scenarios**

The main threats against LTP are the possibility of an LTP engine consuming corrupted segments (either data or control segments) and denial-of-service attacks against a receiving LTP engine.

If an attacker can inject segments into an LTP session, then it can potentially:

- corrupt the data being transferred;

- deny service to an LTP session;
- shut down the session;
- cause unnecessary retransmissions.

To inject LTP segments, the attacker would need to be able to transmit frames using the underlying communication system of the receiving LTP engine and would need to know the LTP session ID (sending engine ID and the session number). Such segments could contain irrelevant or malicious data, or could be LTP control segments such as cancel segments which would shut down ongoing sessions.

To corrupt the data being transferred or to cause unnecessary retransmissions, the attacker would also need to know something about the current state of the LTP session such as whether there is red data in the block, and what data offsets have/have not yet been received.

A denial-of-service attack could be carried out knowing only a valid LTP session ID, where an attacker could initiate multiple LTP sessions with bogus data, causing the receiving LTP engine to store the data waiting for the blocks to complete. Such an attack could consume the available storage at the receiver, preventing legitimate sessions from being established. It should be noted that the receiving LTP engine will eventually, under control of management, time out and remove sessions that do not complete.

An attacker sending malicious report segments could cause unnecessary retransmission by the sender, or could cause the sender to prematurely discard information that had not been acknowledged by the legitimate receiver.

### **D1.3 CONSEQUENCES OF NOT APPLYING SECURITY TO THE TECHNOLOGY**

If an implementation elects to not implement the authentication mechanisms described in section 2.1 of RFC 5327, the system must rely on other layers of the stack for security services, if they are required. If no security were applied at or beneath the LTP layer, then denial-of-service attacks and the sending of corrupt or malicious data to LTP clients would be possible. Even if security were implemented above LTP (e.g., at the Bundle Protocol layer) so that malicious data could not propagate, allowing an unauthenticated third party to inject LTP segments into a communication could allow that party to deny service to legitimate peers of the LTP receiver.

It is strongly recommended that some security measures to prevent injection of malicious data and to prevent denial-of-service attacks be implemented at or below the LTP layer.

## **D2 SANA CONSIDERATIONS**

### **D2.1 LTP ENGINE ID REGISTRY.**

#### **D2.1.1 General**

SANA is requested to establish a registry of CCSDS LTP Engine IDs.

#### **NOTES**

- 1 The purpose of this registry is to ensure uniqueness of LTP Engine IDs that are used in space missions.
- 2 For missions utilizing LTP and BP protocols, requests to SANA should attempt to utilize identical numbers for LTP Protocol Engine Identifiers and BP CBHE Node Numbers.
- 3 The complete space of LTP Engine IDs is managed by IANA (see reference [E5]). The complete space includes values for private / experimental use. A portion of the complete space is delegated to SANA for management, and it is that portion that is described here.

#### **D2.1.2 Value Range for LTP Engine IDs**

The value range for LTP Engine IDs is: integers greater than or equal to 0.

#### **D2.1.3 CCSDS LTP Engine ID Registration Policy**

The registration policy for the registry is: no engineering review required; request must come from the official CCSDS representative of a member agency.

#### **D2.1.4 Initial CCSDS LTP Engine ID Registry**

The initial content of the CCSDS LTP Engine ID Registry is:

**Table D-1: Initial CCSDS LTP Engine ID Registry**

<b>Value</b>	<b>Description</b>	<b>Reference</b>
$2^{14} - 2^{21} - 1$	Unassigned (administered by SANA)	RFC7116, this document



## **D2.2 LTP CLIENT SERVICE ID REGISTRY**

SANA is requested to establish a registry of CCSDS LTP Client Service ID Numbers.

### **NOTES**

- 1 The purpose of this registry is to ensure consistent identification of commonly used LTP clients. This registry is similar to the ‘ethertype’ registry maintained by IANA.
- 2 The complete space of LTP Client Service IDs is managed by IANA (RFC7116, <http://www.iana.org/assignments/ltp-parameters/ltp-parameters.xhtml#client-service-ids>). The complete space includes values for private / experimental use. A portion of the complete space is delegated to SANA for management, and it is that portion that is described here.

In particular, the IANA registry contains entries for the Bundle Protocol (ID 1); the LTP Service Data Aggregation (SDA) service (ID 2); and the CCSDS File Delivery Protocol (ID 3).

### **D2.2.1 Value Range for LTP Client Service ID Numbers**

The value range for LTP Client Service ID Number registry is: integers greater than or equal to 0.

### **D2.2.2 CCSDS LTP Client Service ID Number Registration Policy**

The registration policy for the registry is: change requires a CCSDS approved document.

### **D2.2.3 Initial CCSDS LTP Client Service ID Number Registry**

The initial content of the CCSDS LTP Client Service ID Number Registry is:

**Table D-2: Initial CCSDS LTP Client Service ID Number Registry**

<b>Value</b>	<b>Description</b>	<b>Reference</b>
4 – 16,383	Unassigned (administered by SANA)	RFC7116

## **D2.3 APPLICATION PROTOCOL ID FOR ‘LTP OVER SPACE PACKETS’**

SANA is requested to allocate an application protocol ID for the ‘LTP-for-CCSDS’ protocol from the Space Packet Protocol Application Process Identifier (APID) registry with this document as the reference.

#### **D2.4 PROTOCOL ID ALLOCATION FOR ‘LTP OVER ENCAPSULATION PACKETS’**

SANA is requested to allocate an encapsulation protocol ID for the ‘LTP-for-CCSDS’ protocol from the Protocol Identifiers registry with this document as the reference.

#### **D3 PATENT CONSIDERATIONS**

At the time of publication, CCSDS was not aware of any patents pertaining to the technology described in this document.

## ANNEX E

### INFORMATIVE REFERENCES

#### (INFORMATIVE)

- [E1] K. Scott and S. Burleigh. *Bundle Protocol Specification*. RFC 5050. Reston, Virginia: ISOC, November 2007.
- [E2] S. Burleigh, M. Ramadas, and S. Farrell. *Licklider Transmission Protocol—Motivation*. RFC 5325. Reston, Virginia: ISOC, September 2008.
- [E3] *Rationale, Scenarios, and Requirements for DTN in Space*. Issue 1. Report Concerning Space Data System Standards (Green Book), CCSDS 734.0-G-1. Washington, D.C.: CCSDS, August 2010.
- [E4] W. Eddy and E. Davies. *Using Self-Delimiting Numeric Values in Protocols*. RFC 6256. Reston, Virginia: ISOC, May 2011.
- [E5] K. Scott and M. Blanchet. *Licklider Transmission Protocol (LTP), Compressed Bundle Header Encoding (CBHE), and Bundle Protocol IANA Registries*. RFC 7116. Reston, Virginia: ISOC, February 2014.

## ANNEX F

### ACRONYMS AND ABBREVIATIONS

#### (INFORMATIVE)

<u>Term</u>	<u>Meaning</u>
APID	application process identifier
BP	Bundle Protocol
CFDP	CCSDS File Delivery Protocol
CGBA	commercial generic bioprocessing apparatus
CPU	central processing unit
DCCP	Datagram Congestion Control Protocol
DTN	Delay-Tolerant Networking
ENCAP	CCSDS Encapsulation Service
EOB	end of block
EORP	end of red-part
EPI	Encapsulation Protocol Identifier
IANA	Internet Assigned Numbers Authority
IETF	Internet Engineering Task Force
IUT	implementation under test
LTP	Licklider Transmission Protocol
MIB	management information base
PDU	protocol data unit
PICS	protocol implementation conformance statement
RL	requirements list
RFC	Request for Comments
SANA	Space Assigned Number Authority
SAP	service access point
SDA	service data aggregation
SDNV	self-delimiting numeric value
SDU	service data unit
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
UCP	underlying communication protocols
UDP	User Datagram Protocol