Considerations in Configuring Spans for the Licklider Transmission Protocol

Amalaye Oyake
Flight Software Applications & Data Product Management
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
amalaye.oyake@jpl.nasa.gov

Abstract *– Reliable end-to-end communication in space networks can be provided by Delay Tolerant Networking and its underlying protocols including the Bundle Protocol and the Licklider Transmission Protocol (LTP). LTP is a lower level protocol that tackles the problem of communicating over extremely long distances, and is the focus of this paper. Necessary considerations to be accounted for when configuring the LTP application, as implemented within the Interplanetary Overlay Network software distribution developed by the Jet Propulsion Laboratory, are discussed. LTP enables construction of simple or complex network topologies. The most complicated process involved is often the actualization of data connections establishing one-way communication bridges, or spans, to adjacent network nodes. The concepts outlined in this paper provide guidance through considerations involved in creating space network links using LTP.*

Keywords: Licklider Transmission Protocol, Delay Tolerant Networking, Interplanetary Overlay Network.

1 Introduction

The Delay Tolerant Networking Protocol (DTN), specifically the underlying protocols – Bundle Protocol and the Licklider Transmission Protocol, provide good solutions for enabling reliable end-to-end communication under conditions where persistent end-to-end communication is highly unreliable.

The Interplanetary Overlay Network (ION) software distribution is the implementation developed by the Jet Propulsion Laboratory (JPL) as described in Internet Engineering Task Force RFC 4838 – Delay-Tolerant Networking Architecture [1].

The JPL implementation is designed for use on spacecraft flight hardware, which have strict memory and other resource constraints (power, connectivity, etc). The goal of ION is to simplify the construction of space networks and to enable construction of various network topologies amongst ground, orbital and deep space assets. The ION package consists of the Asynchronous Message Service (AMS), Bundle Protocol (BP) and the Licklider Transmission Protocol (LTP) packages, as well as utility packages and platform abstraction libraries.

The Asynchronous Message Service provides decentralized publish and subscribe capabilities. AMS messages do not go through a central server, rather communication is peer-to-peer, between communicating nodes. Though there is a node registration server, AMS provides protections against single points of failure by providing redundant failover servers in the event of a single registration server failure, in this way application failure can be gracefully handled, without a loss of service.

The Bundle Protocol implements store-and-forward data delivery, custody-based retransmission, the ability to cope with intermittent connectivity, and the ability to take advantage of scheduled, predicted, and opportunistic connectivity. It accomplishes this in two main ways:

1. By storing application data in called bundles and transmitting them when there is a communication opportunity, and
2. By allowing late binding to network endpoints.

The Licklider Transmission Protocol is a lower level protocol, residing underneath the Bundle Protocol. It tackles the problem of communicating over extremely long distances where communication delays incurred from the long one-way light times make traditional Internet protocols unsuitable for typical (undisrupted) communication. LTP is forgiving of these long delays and provides reliable communication through data retransmission as well as an unreliable communication, without data retransmission.

LTP is named after ARPA researcher J. C. R. Licklider, one of the pioneers of ARPANET. Licklider envisioned a concept of devices connected and communicating with one another, even on an interplanetary scale [1, 2].

This paper is primarily concerned with the considerations needed for configuring the LTP application within the ION package.

2 LTP for Space Networking

Under current practice a space network is constructed and deconstructed through laborious efforts, by stringing together each communication node. In such cases, commanding the respective communication node independently configures each node. As previously stated, this process is tedious and time consuming. Furthermore the current network topologies are extremely simplistic and interoperability between different space agencies is difficult.

Thus the use of the DTN protocol standards is a radical deviation from the current practice in spacecraft telecommunications. The goal of DTN is to enable end-to-end networking under conditions where such networks would be impossible with traditional protocols. The expectation is that DTN will enable interoperability across unstable network environments, and improve communication reliability even in the presence of physical constraints and actual disruptions. Interoperability refers to communication between different space assets and different space agencies. In the space community DTN is seen as a tool that can facilitate automatic communication, thereby doing away with much of the current practice of tedious manual configuration.

Some of the difficulties faced in creating end-to-end communication in certain environments are as follows:

- The lack of continuous connections due to occultation by local trusses or occultation by celestial bodies (moons, planet, the sun), orbits, noise and devices being intermittently powered on.
- The lack of interoperability between devices and assets due to heterogeneous data protocols between devices and assets.
- Signal attenuation due to weather conditions.
- Long one-way light times, which cause long time delays.
- Asymmetric communication links (half or full duplex at varying data rates) [3].
- Laborious manual configuration, caused by the individual configuration of devices for every network session.

The set of underlying protocols in the DTN suite specifically address these issues. For instance the Bundle Protocol handles the storage of data until a contact is available to receive it, at which point the data is forwarded. The Bundle Protocol is analogous to the Simple Mail Transfer Protocol used for electronic mail – messages are held for a time, and then forwarded.

Unidirectional and bidirectional communication over very long distances where delays range from tens of seconds to hours is handled at a lower layer by LTP. LTP addresses the aggregation of bundles from the Bundle Protocol into blocks and then the 'chopping up' of these blocks into smaller units called segments, to a size that can fit over the link layer (typically a radio link). LTP also handles reliable (send with acknowledgements) and unreliable (send without acknowledgement) transmission of these segments. The picture of the protocol stack in Figure 1 shows the arrangement of the respective protocol layers.

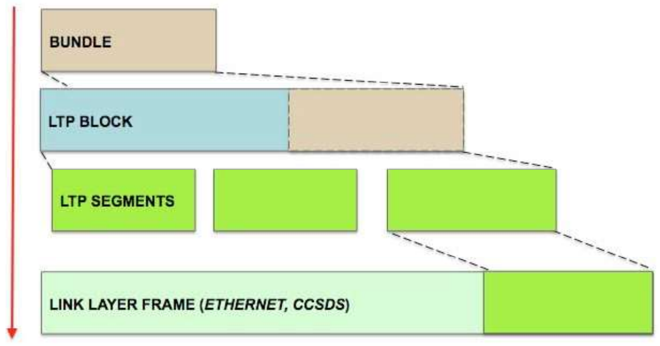

Figure 1. Data layering.

3 Configuration of LTP

Configuring LTP is highly dependent on the scenario and networking environment. Knowledge of the underlying link layer, parameters such as the data rate, the First In, First Out (FIFO) capacities and an understanding of the miscellaneous application queuing delays are very important considerations that must be taken into account. Selecting well-chosen configuration parameters will result in optimal end-to-end communication. Poorly chosen parameters will result in inefficient communication or completely failed end-to-end communication in the worst case.

Prior to configuring LTP, the upper layers of the local DTN protocol are configured (ION, BP and optionally AMS or other applications). These are generic ION configuration applications, establishing communication contacts, in memory databases and Bundle Protocol configuration. These issues will not be discussed in any detail in this paper.

LTP configuration involves establishing one-way communication bridges to adjacent (LTP) nodes as shown in Figure 2. Thus an outbound bridge is established for outgoing traffic and an inbound bridge for inbound traffic. An LTP communication bridge is called a span.

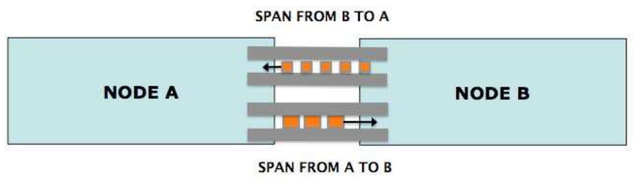

Figure 2. Unidirectional asymmetric spans.

Spans of LTP data interchange are established between the adjacent nodes (e.g., a spacecraft) of the topology and the hub local node (e.g., another spacecraft) [8]. Each span is a point-to-point unidirectional communication link between adjacent pairs of leaf nodes. The return link for each adjacent node is another single unidirectional communication link from the local node to the adjacent leaf node. Each span has its own communication parameters establishing buffer block sizes, segment sizes, data rates and queuing delays.

In this way, various topologies can be constructed by building spans. A relay configuration could look like a star topology for orbital assets and like a tree topology for networks of networks. The complete network stack from the application layer to the data link layer is shown in Figure 3. The physical layer for space communications is via radio links, thus we are concerned with using LTP for communicating over wireless data links.

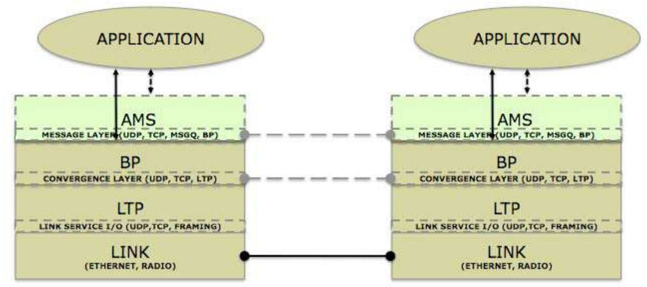
APPLICATION
APPLICATION
AMS
MESSAGE LAYER (UDP, TCP, MSGQ, BP)
BP
CONVERGENCE LAYER (UDP, TCP, LTP)
LTP
LINK SERVICE I/O (UDP,TCP, FRAMING)
LINK
(ETHERNET, RADIO)
AMS
MESSAGE LAYER (UDP, TCP, MSGQ, BP)
BP
CONVERGENCE LAYER (UDP, TCP, LTP)
LTP
LINK SERVICE I/O (UDP,TCP, FRAMING)
LINK
(ETHERNET, RADIO)

Figure 3. Protocol Stack.

The following is a list of required LTP configuration parameters for constructing a span:

- MAX_EXPORT_SESSIONS
- MAX_EXPORT_SESSION_BLOCK_SIZE
- MAX_IMPORT_SESSIONS
- MAX_IMPORT_SESSION_BLOCK_SIZE
- MAX_SEGMENT_SIZE
- AGGREGATION_SIZE_LIMIT
- AGGREGATION_TIME_LIMIT

MAX_EXPORT_SESSIONS: This parameter states the maximum number of concurrent connections to a node, an important load balancing parameter, as buffer space on the node must be strictly managed and a node does not have the liberty to accept unlimited connections [8].

MAX_EXPORT_SESSION_BLOCK_SIZE: This parameter specifies the size in bytes of the blocks that LTP generates. A block should be larger than the size of the inbound bundles generated by the Bundle Protocol service. An upstream Bundle Protocol Service will nominally establish the size of its outbound bundles and the outbound data rate. These outbound bundles are fed into the local LTP service for consumption. LTP aggregates the bundles into blocks. This parameter specifies the maximum LTP block size that will contain the bundles. It should be appropriately sized and consistent with the parameters established by this upstream Bundle Protocol Service.

MAX_IMPORT_SESSIONS: This parameter states the maximum number of concurrent connections to the neighboring node. The neighboring node may have very different connection limitations [8].

MAX_IMPORT_SESSION_BLOCK_SIZE: This parameter states the maximum block size of the local (receiving) node. That is the maximum block size an adjacent node can send to the local node [8].

MAX_SEGMENT_SIZE: This is one of the most important parameters. Blocks are broken down into smaller units called segments. When segments are transmitted they are reconstructed on the receiving end and transformed into blocks, from which the bundles are extracted. The segment size must be small enough to fit into the FIFO buffer of the lower link service, typically a radio, but could even be an Ethernet or serial connection [8].

AGGREGATION_SIZE_LIMIT: This parameter limits the size of a block and constrains the number of bundles that can be aggregated into a block. When the block is filled with bundles it is dispatched for segmentation and transmission [8].

AGGREGATION_TIME_LIMIT: A timer constrains the aggregation of bundles into blocks, so that a block will not wait forever before being released for segmentation and transmission [8].

In the ION package LTP is configured using the ltpadmin utility. Normally all of the ION commands are run in a script as shown below.

```
ionadmin marsrover.ionrc #Configure ION

ionadmin global.ionrc #Configure the contacts

bpadmin marsrover.bprc #Configure bundle protocol

ltpadmin marsrover.ltprc #Configure LTP
```

An example LTP configuration file is shown below.

```
# LTP Configuration File
1 2 4000000
a span 1 2 500000 2 500000 1000 500000 1 'udplso
192.168.10.10:15001'
s 'udplsi 192.168.10.11:15001'
m screening n
m ownqtime  2
w 1
l span
```

The format of the configuration parameters for building a span is as follows:

```
1 est_max_export_sessions
database_bytes_needed

a span peer_engine_nbr
max_export_sessions
max_export_session_block_size
max_import_sessions
max_import_session_block_size
max_segment_size aggregation_size_limit
aggregation_time_limit 'LSO_command'
[queuing_latency]

s 'LSI command'
```

So, the following initializes LTP with a maximum of 2 export sessions and 4000000 bytes for the local LTP engine:

```
1 2 4000000
```

This is the first command provided to LTP. LTP cannot be used until it is initialized.

The following creates a one-directional data communication bridge between the local LTP engine and a remote LTP engine (numbered 2):

```
a span 2 10 500000 2 500000 1000 500000
1 'udplso 137.79.30.214:15001' 4
```

With respect to this span:

- There are a maximum of 10 concurrent connections from this node to other nodes.
- The maximum size of any single bundle that can be sent by this node is 500000 bytes.
- The local node can have a maximum of 2 concurrent connections to it from other nodes.
- The maximum size of any single bundle sent *to* this node is 50000 bytes.
- The maximum size of any single bundle sent *from* this node is 50000 bytes.
- The maximum segment size for this span is set to 1000 bytes.
- The aggregation size limit is 500000 bytes.
- The aggregation time limit is 2 seconds. Thus after 2 seconds the LTP engine will dispatch the current block.
- A link service output task is spawned at the end of this command. In this case, the span created is a User Datagram Protocol (UDP) bridge to node 2, by which LTP segments travel.
- There is a queuing latency of 4 seconds to account for miscellaneous system queuing delays.

The segments are sized appropriately so that the segment data fits on the underlying transport mechanism, which under operational cases is a radio link as shown in Figure 4. Thus a span establishes a bridge for the output of LTP segments from the local node.

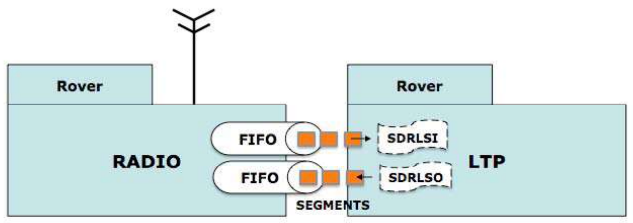

Figure 4. Feeding a radio with LTP segments.

Incoming LTP segments are received by spawning a task that receives these segments from the span of the adjacent node. The following command accomplishes this:

```
s 'udplsi 192.168.10.11:15001'
```

It starts a listener task that will receive LTP segments from the IP address and port number specified.

It should be noted that when using actual radios, the specific application to send and receive data would be different. Such an application can be easily tailored to support the specific radio hardware. One could envision applications called *sdrlso* and *sdrlsi*, which handle the sending and receiving of LTP segments to a radio as shown in Figure 4.

An example of this configuration is shown below.

```
# Software defined radio out
a span 1 2 5000 2 5000 1000 5000 1 'sdrlso'

# Software defined radio in
s 'sdrlsi'
```

An example of an operational scenario is autonomous relay communication using LTP as shown in Figure 5. In this scenario an orbiter communicates with a rover. The orbiter and rover normally communicate using UHF frequencies and utilize the CCSDS Proximity-1 Space Link Protocol (http://www.ccsds.org/) to automatically establish a handshake.

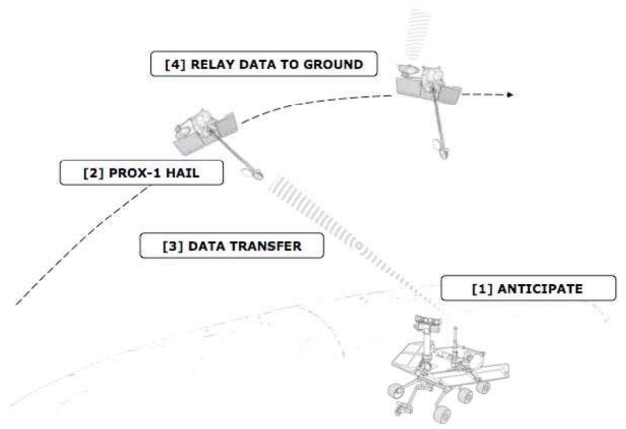

Figure 5. Autonomous relay operations.

Once the handshake is established, communication can occur by a command sequence provided by ground controllers. However it would be more efficient if the human-in-the-loop is removed and the communication is automatically initiated. This is where DTN plays a role. Given the availability of a contact (orbiter to rover and rover to orbiter) the spacecraft and the rover initiate communication and the rover relays its data to the orbiter.

LTP segments would be transmitted as data elements of the CCSDS Proximity-1 protocol. The orbiter would receive this telemetry and reconstruct the DTN bundles from the segments, or simply forward the segments to the final destination, which would normally be a ground station.

Note that LTP acknowledgements could be flowing in the opposite direction from the orbiter to the spacecraft if the link is a full-duplex communication link and LTP reliability is used. These acknowledgements are also segments encapsulated in Proximity-1 data. Data is then automatically purged from the rover and a copy is not kept for retransmission, which is a common practice.

A software architecture for a spacecraft using DTN services to perform relay operations can be envisioned by the UML package diagram shown in Figure 6. The subsystems communicate by exchanging messages between their various interfaces. Arrows between the subsystems show the high-level interactions.

The management of data products resident on the file system (not shown) is handled by the DTN subsystem, in this example the JPL ION implementation of DTN. ION communicates with the Communication Subsystem by encapsulating LTP segments in a frame format (CCSDS or others). The Communication Subsystem relies on ancillary information to establish the communication link. The data link is maintained through the protocol handshake at the DTN layer. Nominally LTP segments will be sent and these segments are acknowledged by sending an acknowledgement message to the sending node.

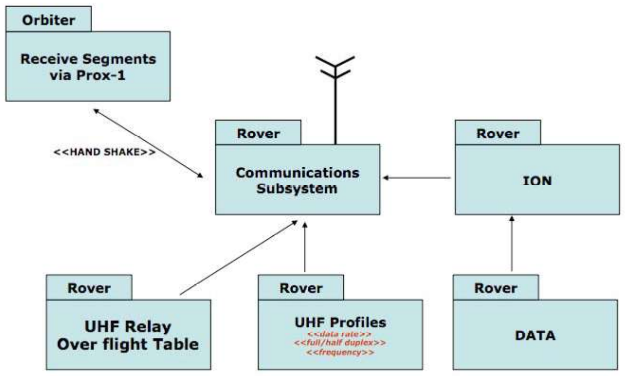

Figure 6. High-level software architecture.

4 Conclusion

LTP is a very important part of the DTN protocol stack that addresses the problem of reliable communication by retransmitting the unacknowledged message parts (the segments).

Using LTP, simple or complex network topologies can be constructed. These networks links look like dual bridges between adjacent nodes, which exchange LTP segments.

At a higher level the Bundle Protocol and the ION administrative tools address issues such as node addressing, store-and-forward data handling, and data routing. The actualization of the data connection is often the most complicated process because it may involve interfacing LTP to a low level device such as a radio.

It should be noted that DTN has already been successfully demonstrated on two spacecraft, the United Kingdom-Disaster Monitoring Constellation satellite (UK-DMC) [9] and the Disruption Tolerant Networking Flight Validation Experiment (DINET) using the Deep Impact spacecraft [10]. The use of DTN extends to terrestrial applications as well [11].

The concepts outlined should provide some guidance in the considerations that should be made in creating these network links.

Acknowledgment

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] M. Ramadas, "LTP & A Priority– Paradigm for deep-SPACE applications," 2nd IEEE International Conference on Space Mission Challenges for Information Technology, Pasadena, CA, 2006.

[2] J.C.R. Licklider, "Memorandum for members and affiliates of the intergalactic computer network," RG 330-69-A-4998, Box 3, Folder: 350-1, National Archives and Records Administration, Washington D.C., April 23, 1963.

[3] S. Burleigh, M. Ramadas and S. Farrell, "Licklider Transmission Protocol – Motivation," Delay Tolerant Networking Research Group, IETF RFC 5325, September 2008, http://www.ietf.org/rfc/rfc5325.txt.

[4] M. Ramadas, S. Burleigh and S. Farrell, "Licklider Transmission Protocol – Specification," Delay Tolerant Networking Research Group, IETF RFC 5326, September 2008, http://www.ietf.org/rfc/rfc5326.txt.

[5] S. Burleigh, H. Kruse and S. Farrell, "Licklider Transmission Protocol (LTP): An overview," August 2004, http://www.dtnrg.org/docs/presentations/IETF60/LTPSanDiegoAll.pdf.

[6] S. Farrell and V. Cahill, "LTP-T: A generic delay tolerant transport protocol," Technical Report, TCD-CS-2005-69, Distributed Systems Group, Department of Computer Science, Trinity College Dublin, Ireland, 2005.

[7] S. Burleigh, "A Guide to Configuring LTP for ION," Jet Propulsion Laboratory, California Institute of Technology, June 10, 2009.

[8] S. Burleigh, *Interplanetary Overlay Network (ION) Design and Operation*, V1.8, JPL D-48259, Jet Propulsion Laboratory, California Institute of Technology, Feb. 6, 2008, https://ion.ocp.ohiou.edu/legacy/ION.pdf.

[9] W. Ivancic, W.M. Eddy, L. Wood, D. Stewart, C. Jackson, J. Northam and A. da Silva Curiel, "Delay/disruption-tolerant network testing using a LEO satellite," Proc. NASA Earth Science Technology Conference, College Park, MD, June 24-26, 2008.

[10] J. Wyatt, S. Burleigh, R. Jones, L. Torgerson and S. Wissler, "Disruption tolerant networking flight validation experiment on NASA's EPOXI mission," Proc. First International Conference on Advances in Satellite and Space Communications, Colmar, France, pp. 187-196, 2009.

[11] K. Fall, "A delay-tolerant network architecture for challenged Internets," Proc. ACM annual conference of the Special Interest Group on Data Communication (SIGCOMM), Karlsruhe, Germany, Aug. 2003.