       ION Licklider Transmission Protocol Admin Application Data Model
                   draft-birrane-dtn-adm-ion-ltpadmin-01

Abstract

   This document describes the Application Data Model (ADM) for the
   configuration of licklider transmission protocol (LTP) in ION in
   compliance with the template provided by [I-D.birrane-dtn-adm].

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   An Application Data Model (ADM) provides a guaranteed interface for
   the management of an application or protocol in accordance with the
   Asynchronous Management Architecture (AMA) defined in
   [I-D.birrane-dtn-ama]. The ADM described in this document complies
   with the ADM Template provided in [I-D.birrane-dtn-adm] as encoded
   using the JSON syntax.

   The ION Licklider Transmission Protocol (LTP) Administration ADM
   contains all of the functionality that is required to properly
   configure LTP in ION in accordance with [RFC5326]. LTP is a
   convergence layer protocol. There is no flow control or congestion
   control in LTP. LTP must run either over UDP or directly over a link
   layer protocol. Because of this, LTP cannot be used in every
   situation. This LTP Admin ADM provides the set of information
   necessary to provide retransmission based reliability on challenged
   networks, focusing on the information that are needed to manage LTP
   properly on the network.

1.1. Technical Notes

   o This document describes Version 0.0 of the ION LTP Admin ADM.

   o The AMM Resource Identifier (ARI) for this ADM is NOT correctly
      set. A sample ARI is used in this version of the specification
      and MAY change in future versions of this ADM until an ARI
      registry is established. This notice will be removed at that
      time.

   o Agent applications MAY choose to ignore the name, description, or
      other annotative information associated with the component

definitions within this ADM where such items are only used to
provide human-readable information or are otherwise not necessary
to manage a device.

1.2. Scope

This ADM specifies those components of the Asynchronous Management
Model (AMM) common to the configuration of LTP in ION.

Any Manager software implementing this ADM MUST perform the
responsibilities of an AMA Manager as outlined in
[I-D.birrane-dtn-adm] as they relate to the objects included in this
document.

Any Agent software implementing this ADM MUST perform the
responsibilities of an AMA Agent as outlined in [I-D.birrane-dtn-adm]
as they relate to the objects included in this document.

1.3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in RFC 2119 [RFC2119].

2. Structure and Design of this ADM

The LTP Admin ADM's structure is in accordance to
[I-D.birrane-dtn-adm]. This ADM contains metadata, table templates,
and controls. Table Templates are column templates that will be
followed by any instance of this table available in the network.
They may not be created dynamically within the network by Managers.
Controls are predefined and sometimes parameterized opcodes that can
be run on an Agent. Controls are preconfigured in Agents and
Managers as part of ADM support.There are no variables, report
templates, macros, edd, constants, or operators in this ADM at this
time. The contents of this ADM are derived from the main functions
and data that are needed to manage LTP RFC 5326 [RFC5326].

All ADMs have metadata that includes the name, namespace, and version
of the ADM as well as the name of the organization that is issuing
that particular ADM. This is important for identification purposes
of the ADMs and to ensure version control.

The main idea identified in LTP that is present in this ADM is a span
of potential LTP data interchange between engines(nodes on a network
that use LTP).

3. Naming and Identification

   This section outlines the namespaces used to uniquely identify ADM
   objects in this specification.

3.1. Namespace and Nicknames

   In accordance with [I-D.birrane-dtn-adm], every ADM is assigned a
   moderated Namespace. In accordance with [I-D.birrane-dtn-amp], these
   namespaces may be enumerated for compactness. The namespace and ADM
   identification for these objects is defined as follows.

```
                +-----------------+------------------+
                |    Identifier   |      Value       |
                +-----------------+------------------+
                |    Namespace    | DTN/ION/ltpadmin |
                |                 |                  |
                | ADM Enumeration |        9         |
                +-----------------+------------------+
```

                   Table 1: Namespace Information

   Given the above ADM enumeration, in accordance with
   [I-D.birrane-dtn-amp], the following AMP nicknames are defined.

```
            +----------+-----------------------------+
            | Nickname |          Collection         |
            +----------+-----------------------------+
            |   180    |    DTN/ION/ltpadmin//Const   |
            |          |                             |
            |   181    |     DTN/ION/ltpadmin//Ctrl   |
            |          |                             |
            |   182    |     DTN/ION/ltpadmin//Edd    |
            |          |                             |
            |   183    |     DTN/ION/ltpadmin//Mac    |
            |          |                             |
            |   184    |     DTN/ION/ltpadmin//Oper   |
            |          |                             |
            |   185    |     DTN/ION/ltpadmin//Rptt   |
            |          |                             |
            |   187    |     DTN/ION/ltpadmin//Tblt   |
            |          |                             |
            |   189    |     DTN/ION/ltpadmin//Var    |
            |          |                             |
            |   190    |     DTN/ION/ltpadmin//Mdat   |
            |          |                             |
            | 191-199  |  DTN/ION/ltpadmin//Reserved |
            +----------+-----------------------------+
```

Table 2: ION BP ADM Nicknames

4. ION LTP Admin ADM JSON Encoding

```
 {
   "Mdat": [{
       "name": "name",
       "type": "STR",
       "value": "ion_ltp_admin",
       "description": "The human-readable name of the ADM."
     },
     {
       "name": "namespace",
       "type": "STR",
       "value": "DTN/ION/ltpadmin",
       "description": "The namespace of the ADM."
     },
     {
       "name": "version",
       "type": "STR",
       "value": "v0.0",
       "description": "The version of the ADM."
     },
```

```
      {
        "name": "organization",
        "type": "STR",
        "value": "JHUAPL",
        "description": "The name of the issuing organization of the
                        ADM."
      }
    ],
    "Edd": [{
      "name": "ion_version",
      "type": "STR",
      "description": "This is the version of ION that is currently
                      installed."
    }],
    "Tblt": [{
      "name": "spans",
      "columns": [{
        "type": "UVAST",
        "name": "peer_engine_nbr"
      }, {
        "type": "UINT",
        "name": "max_export_sessions"
      }, {
        "type": "UINT",
        "name": "max_import_sessions"
      }, {
        "type": "UINT",
        "name": "max_segment_size"
      }, {
        "type": "UINT",
        "name": "aggregation_size_limit"
      }, {
        "type": "UINT",
        "name": "aggregation_time_limit"
      }, {
        "type": "STR",
        "name": "lso_control"
      }, {
        "type": "UINT",
        "name": "queueing_latency"
      }],
      "description": "This table lists all spans of potential LTP data
                      interchange that exists between the local LTP
                      engine and the indicated (neighboring) LTP engine."
    }],
    "Ctrl": [{
        "name": "manage_heap",
        "parmspec": [{
```

```
      "type": "UINT",
      "name": "max_database_heap_per_block"
    }],
    "description": "This control declares the maximum number of
                    bytes of SDR heap space that will be occupied by
                    the acquisition of any single LTP block. All
                    data acquired in excess of this limit will be
                    written to a temporary file pending extraction
                    and dispatching of the acquired block. Default
                    is the minimum allowed value (560 bytes), which
                    is the approximate size of a ZCO file reference
                    object; this is the minimum SDR heap space
                    occupancy in the event that all acquisition is
                    into a file."
  },
  {
    "name": "manage_max_ber",
    "parmspec": [{
      "type": "REAL32",
      "name": "max_expected_bit_error_rate"
    }],
    "description": "This control sets the expected maximum bit error
                    rate(BER) that LTP should provide for in
                    computing the maximum number of transmission
                    efforts to initiate in the transmission of a
                    given block.(Note that this computation is also
                    sensitive to data segment size and to the size
                    of the block that is to be transmitted.) The
                    default value is .0001 (10^-4)."
  },
  {
    "name": "manage_own_queue_time",
    "parmspec": [{
      "type": "UINT",
      "name": "own_queing_latency"
    }],
    "description": "This control sets the number of seconds of
                    predicted additional latency attributable to
                    processing delay within the local engine itself
                    that should be included whenever LTP computes
                    the nominal round-trip time for an exchange of
                    data with any remote engine.The default value
                    is 1."
  },
  {
    "name": "manage_screening",
    "parmspec": [{
      "type": "UINT",
```

```
            "name": "new_state"
        }],
        "description": "This control enables or disables the screening
                        of received LTP segments per the periods of
                        scheduled reception in the node's contact graph.
                        By default, screening is disabled. When
                        screening is enabled, such segments are silently
                        discarded. Note that when screening is enabled
                        the ranges declared in the contact graph must
                        be accurate and clocks must be synchronized;
                        otherwise, segments will be arriving at times
                        other than the scheduled contact intervals and
                        will be discarded."
    },
    {
        "name": "span_add",
        "parmspec": [{
            "type": "UVAST",
            "name": "peer_engine_number"
        }, {
            "type": "UINT",
            "name": "max_export_sessions"
        }, {
            "type": "UINT",
            "name": "max_import_sessions"
        }, {
            "type": "UINT",
            "name": "max_segment_size"
        }, {
            "type": "UINT",
            "name": "aggregtion_size_limit"
        }, {
            "type": "UINT",
            "name": "aggregation_time_limit"
        }, {
            "type": "STR",
            "name": "lso_control"
        }, {
            "type": "UINT",
            "name": "queuing_latency"
        }],
        "description": "This control declares that a span of potential
                        LTP data interchange exists between the local
                        LTP engine and the indicated (neighboring) LTP
                        engine."
    },
    {
        "name": "span_change",
```

```
      "parmspec": [{
        "type": "UVAST",
        "name": "peer_engine_number"
      }, {
        "type": "UINT",
        "name": "max_export_sessions"
      }, {
        "type": "UINT",
        "name": "max_import_sessions"
      }, {
        "type": "UINT",
        "name": "max_segment_size"
      }, {
        "type": "UINT",
        "name": "aggregtion_size_limit"
      }, {
        "type": "UINT",
        "name": "aggregation_time_limit"
      }, {
        "type": "STR",
        "name": "lso_control"
      }, {
        "type": "UINT",
        "name": "queuing_latency"
      }],
      "description": "This control sets the indicated span's
                      configuration parameters to the values provided
                      as arguments"
    },
    {
      "name": "span_del",
      "parmspec": [{
        "type": "UVAST",
        "name": "peer_engine_number"
      }],
      "description": "This control deletes the span identified by
                      peerEngineNumber. The control will fail if any
                      outbound segments for this span are pending
                      transmission or any inbound blocks from the
                      peer engine are incomplete."
    },
    {
      "name": "stop",
      "description": "This control stops all link service input and
                      output tasks for the local LTP engine."
    },
    {
      "name": "watch_set",
```

```
      "parmspec": [{
        "type": "STR",
        "name": "activity"
      }],
      "description": "This control enables and disables production of
      a continuous stream of user- selected LTP activity indication
      characters. Activity parameter of 1 selects all LTP activity
      indication characters; 0 de-selects all LTP activity indication
      characters; any other activitySpec such as df{] selects all
      activity indication characters in the string, de-selecting all
      others. LTP will print each selected activity indication
      character to stdout every time a processing event of the
      associated type occurs: d bundle appended to block for next
      session, e segment of block is queued for transmission, f
      block has been fully segmented for transmission, g segment
      popped from transmission queue, h positive ACK received for
      block and session ended, s segment received, t block has been
      fully received, @ negative ACK received for block and segments
      retransmitted, = unacknowledged checkpoint was retransmitted,
      + unacknowledged report segment was retransmitted, { export
      session canceled locally (by sender), } import session canceled
      by remote sender, [ import session canceled locally (by
      receiver), ] export session canceled by remote receiver"
    }
  ]
 }
```

5. IANA Considerations

   At this time, this protocol has no fields registered by IANA.

6. References

6.1. Informative References

   [I-D.birrane-dtn-ama]
            Birrane, E., "Asynchronous Management Architecture",
            draft-birrane-dtn-ama-07 (work in progress), June 2018.

6.2. Normative References

   [I-D.birrane-dtn-adm]
            Birrane, E., DiPietro, E., and D. Linko, "AMA Application
            Data Model", draft-birrane-dtn-adm-02 (work in progress),
            June 2018.

   [I-D.birrane-dtn-amp]
              Birrane, E., "Asynchronous Management Protocol", draft-
              birrane-dtn-amp-04 (work in progress), June 2018.

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119,
              DOI 10.17487/RFC2119, March 1997,
              <https://www.rfc-editor.org/info/rfc2119>.

   [RFC5326] Ramadas, M., Burleigh, S., and S. Farrell, "Licklider
              Transmission Protocol - Specification", RFC 5326,
              DOI 10.17487/RFC5326, September 2008,
              <https://www.rfc-editor.org/info/rfc5326>.

Authors' Addresses

   Edward J. Birrane
   Johns Hopkins Applied Physics Laboratory

   Email: Edward.Birrane@jhuapl.edu

   Evana DiPietro
   Johns Hopkins Applied Physics Laboratory

   Email: Evana.DiPietro@jhuapl.edu

   David Linko
   Johns Hopkins Applied Physics Laboratory

   Email: David.Linko@jhuapl.edu