Network Working Group   V. Cerf
Request for Comments: 4838   Google/Jet Propulsion Laboratory
Category: Informational   S. Burleigh
                                                             A. Hooke
                                                         L. Torgerson
                                         NASA/Jet Propulsion Laboratory
                                                             R. Durst
                                                             K. Scott
                                                   The MITRE Corporation
                                                              K. Fall
                                                    Intel Corporation
                                                             H. Weiss
                                                         SPARTA, Inc.
                                                           April 2007

                    Delay-Tolerant Networking Architecture

Status of This Memo

Copyright Notice

IESG Note

Abstract

   This document describes an architecture for delay-tolerant and
   disruption-tolerant networks, and is an evolution of the architecture
   originally designed for the Interplanetary Internet, a communication
   system envisioned to provide Internet-like services across
   interplanetary distances in support of deep space exploration. This
   document describes an architecture that addresses a variety of
   problems with internetworks having operational and performance
   characteristics that make conventional (Internet-like) networking
   approaches either unworkable or impractical. We define a message-
   oriented overlay that exists above the transport (or other) layers of

the networks it interconnects. The document presents a motivation
for the architecture, an architectural overview, review of state
management required for its operation, and a discussion of
application design issues. This document represents the consensus of
the IRTF DTN research group and has been widely reviewed by that
group.

Table of Contents

1. Introduction

   This document describes an architecture for delay and disruption-
   tolerant interoperable networking (DTN). The architecture embraces
   the concepts of occasionally-connected networks that may suffer from
   frequent partitions and that may be comprised of more than one
   divergent set of protocols or protocol families. The basis for this
   architecture lies with that of the Interplanetary Internet, which
   focused primarily on the issue of deep space communication in high-
   delay environments. We expect the DTN architecture described here to
   be utilized in various operational environments, including those
   subject to disruption and disconnection and those with high-delay;
   the case of deep space is one specialized example of these, and is
   being pursued as a specialization of this architecture (See [IPN01]
   and [SB03] for more details).

   Other networks to which we believe this architecture applies include
   sensor-based networks using scheduled intermittent connectivity,
   terrestrial wireless networks that cannot ordinarily maintain end-to-
   end connectivity, satellite networks with moderate delays and
   periodic connectivity, and underwater acoustic networks with moderate
   delays and frequent interruptions due to environmental factors. A
   DTN tutorial [FW03], aimed at introducing DTN and the types of
   networks for which it is designed, is available to introduce new
   readers to the fundamental concepts and motivation. More technical
   descriptions may be found in [KF03], [JFP04], [JDPF05], and [WJMF05].

   We define an end-to-end message-oriented overlay called the "bundle
   layer" that exists at a layer above the transport (or other) layers
   of the networks on which it is hosted and below applications.
   Devices implementing the bundle layer are called DTN nodes. The
   bundle layer forms an overlay that employs persistent storage to help
   combat network interruption. It includes a hop-by-hop transfer of
   reliable delivery responsibility and optional end-to-end
   acknowledgement. It also includes a number of diagnostic and
   management features. For interoperability, it uses a flexible naming
   scheme (based on Uniform Resource Identifiers [RFC3986]) capable of
   encapsulating different naming and addressing schemes in the same
   overall naming syntax. It also has a basic security model,
   optionally enabled, aimed at protecting infrastructure from
   unauthorized use.

   The bundle layer provides functionality similar to the internet layer
   of gateways described in the original ARPANET/Internet designs
   [CK74]. It differs from ARPANET gateways, however, because it is
   layer-agnostic and is focused on virtual message forwarding rather
   than packet switching. However, both generally provide
   interoperability between underlying protocols specific to one

environment and those protocols specific to another, and both provide
a store-and-forward forwarding service (with the bundle layer
employing persistent storage for its store and forward function).

In a sense, the DTN architecture provides a common method for
interconnecting heterogeneous gateways or proxies that employ store-
and-forward message routing to overcome communication disruptions.
It provides services similar to electronic mail, but with enhanced
naming, routing, and security capabilities. Nodes unable to support
the full capabilities required by this architecture may be supported
by application-layer proxies acting as DTN applications.

2. Why an Architecture for Delay-Tolerant Networking?

Our motivation for pursuing an architecture for delay tolerant
networking stems from several factors. These factors are summarized
below; much more detail on their rationale can be explored in [SB03],
[KF03], and [DFS02].

The existing Internet protocols do not work well for some
environments, due to some fundamental assumptions built into the
Internet architecture:

- that an end-to-end path between source and destination exists for
  the duration of a communication session

- (for reliable communication) that retransmissions based on timely
  and stable feedback from data receivers is an effective means for
  repairing errors

- that end-to-end loss is relatively small

- that all routers and end stations support the TCP/IP protocols

- that applications need not worry about communication performance

- that endpoint-based security mechanisms are sufficient for meeting
  most security concerns

- that packet switching is the most appropriate abstraction for
  interoperability and performance

- that selecting a single route between sender and receiver is
  sufficient for achieving acceptable communication performance

The DTN architecture is conceived to relax most of these assumptions,
based on a number of design principles that are summarized here (and
further discussed in [KF03]):

- Use variable-length (possibly long) messages (not streams or
  limited-sized packets) as the communication abstraction to help
  enhance the ability of the network to make good scheduling/path
  selection decisions when possible.

- Use a naming syntax that supports a wide range of naming and
  addressing conventions to enhance interoperability.

- Use storage within the network to support store-and-forward
  operation over multiple paths, and over potentially long timescales
  (i.e., to support operation in environments where many and/or no
  end-to-end paths may ever exist); do not require end-to-end
  reliability.

- Provide security mechanisms that protect the infrastructure from
  unauthorized use by discarding traffic as quickly as possible.

- Provide coarse-grained classes of service, delivery options, and a
  way to express the useful lifetime of data to allow the network to
  better deliver data in serving the needs of applications.

The use of the bundle layer is guided not only by its own design
principles, but also by a few application design principles:

- Applications should minimize the number of round-trip exchanges.

- Applications should cope with restarts after failure while network
  transactions remain pending.

- Applications should inform the network of the useful life and
  relative importance of data to be delivered.

These issues are discussed in further detail in Section 5.

3. DTN Architectural Description

   The previous section summarized the design principles that guide the
   definition of the DTN architecture. This section presents a
   description of the major features of the architecture resulting from
   design decisions guided by the aforementioned design principles.

3.1. Virtual Message Switching Using Store-and-Forward Operation

   A DTN-enabled application sends messages of arbitrary length, also
   called Application Data Units or ADUs [CT90], which are subject to
   any implementation limitations. The relative order of ADUs might not
   be preserved. ADUs are typically sent by and delivered to

applications in complete units, although a system interface that
behaves differently is not precluded.

ADUs are transformed by the bundle layer into one or more protocol
data units called "bundles", which are forwarded by DTN nodes.
Bundles have a defined format containing two or more "blocks" of
data. Each block may contain either application data or other
information used to deliver the containing bundle to its
destination(s). Blocks serve the purpose of holding information
typically found in the header or payload portion of protocol data
units in other protocol architectures. The term "block" is used
instead of "header" because blocks may not appear at the beginning of
a bundle due to particular processing requirements (e.g., digital
signatures).

Bundles may be split up ("fragmented") into multiple constituent
bundles (also called "fragments" or "bundle fragments") during
transmission. Fragments are themselves bundles, and may be further
fragmented. Two or more fragments may be reassembled anywhere in the
network, forming a new bundle.

Bundle sources and destinations are identified by (variable-length)
Endpoint Identifiers (EIDs, described below), which identify the
original sender and final destination(s) of bundles, respectively.
Bundles also contain a "report-to" EID used when special operations
are requested to direct diagnostic output to an arbitrary entity
(e.g., other than the source). An EID may refer to one or more DTN
nodes (i.e., for multicast destinations or "report-to" destinations).

While IP networks are based on "store-and-forward" operation, there
is an assumption that the "storing" will not persist for more than a
modest amount of time, on the order of the queuing and transmission
delay. In contrast, the DTN architecture does not expect that
network links are always available or reliable, and instead expects
that nodes may choose to store bundles for some time. We anticipate
that most DTN nodes will use some form of persistent storage for this
-- disk, flash memory, etc. -- and that stored bundles will survive
system restarts.

Bundles contain an originating timestamp, useful life indicator, a
class of service designator, and a length. This information provides
bundle-layer routing with a priori knowledge of the size and
performance requirements of requested data transfers. When there is
a significant amount of queuing that can occur in the network (as is
the case in the DTN version of store-and-forward), the advantage
provided by knowing this information may be significant for making
scheduling and path selection decisions [JFP04]. An alternative
abstraction (i.e., of stream-based delivery based on packets) would

make such scheduling much more difficult. Although packets provide
some of the same benefits as bundles, larger aggregates provide a way
for the network to apply scheduling and buffer management to units of
data that are more useful to applications.

An essential element of the bundle-based style of forwarding is that
bundles have a place to wait in a queue until a communication
opportunity ("contact") is available. This highlights the following
assumptions:

1. that storage is available and well-distributed throughout the
   network,

2. that storage is sufficiently persistent and robust to store
   bundles until forwarding can occur, and

3. (implicitly) that this "store-and-forward" model is a better
   choice than attempting to effect continuous connectivity or other
   alternatives.

For a network to effectively support the DTN architecture, these
assumptions must be considered and must be found to hold. Even so,
the inclusion of long-term storage as a fundamental aspect of the DTN
architecture poses new problems, especially with respect to
congestion management and denial-of-service mitigation. Node storage
in essence represents a new resource that must be managed and
protected. Much of the research in DTN revolves around exploring
these issues. Congestion is discussed in Section 3.13, and security
mechanisms, including methods for DTN nodes to protect themselves
from handling unauthorized traffic from other nodes, are discussed in
[DTNSEC] and [DTNSOV].

3.2. Nodes and Endpoints

A DTN node (or simply "node" in this document) is an engine for
sending and receiving bundles -- an implementation of the bundle
layer. Applications utilize DTN nodes to send or receive ADUs
carried in bundles (applications also use DTN nodes when acting as
report-to destinations for diagnostic information carried in
bundles). Nodes may be members of groups called "DTN endpoints". A
DTN endpoint is therefore a set of DTN nodes. A bundle is considered
to have been successfully delivered to a DTN endpoint when some
minimum subset of the nodes in the endpoint has received the bundle
without error. This subset is called the "minimum reception group"
(MRG) of the endpoint. The MRG of an endpoint may refer to one node
(unicast), one of a group of nodes (anycast), or all of a group of
nodes (multicast and broadcast). A single node may be in the MRG of
multiple endpoints.

3.3. Endpoint Identifiers (EIDs) and Registrations

   An Endpoint Identifier (EID) is a name, expressed using the general
   syntax of URIs (see below), that identifies a DTN endpoint. Using an
   EID, a node is able to determine the MRG of the DTN endpoint named by
   the EID. Each node is also required to have at least one EID that
   uniquely identifies it.

   Applications send ADUs destined for an EID, and may arrange for ADUs
   sent to a particular EID to be delivered to them. Depending on the
   construction of the EID being used (see below), there may be a
   provision for wildcarding some portion of an EID, which is often
   useful for diagnostic and routing purposes.

   An application's desire to receive ADUs destined for a particular EID
   is called a "registration", and in general is maintained persistently
   by a DTN node. This allows application registration information to
   survive application and operating system restarts.

   An application's attempt to establish a registration is not
   guaranteed to succeed. For example, an application could request to
   register itself to receive ADUs by specifying an Endpoint ID that is
   uninterpretable or unavailable to the DTN node servicing the request.
   Such requests are likely to fail.

3.3.1. URI Schemes

   Each Endpoint ID is expressed syntactically as a Uniform Resource
   Identifier (URI) [RFC3986]. The URI syntax has been designed as a
   way to express names or addresses for a wide range of purposes, and
   is therefore useful for constructing names for DTN endpoints.

   In URI terminology, each URI begins with a scheme name. The scheme
   name is an element of the set of globally-managed scheme names
   maintained by IANA [ISCHEMES]. Lexically following the scheme name
   in a URI is a series of characters constrained by the syntax defined
   by the scheme. This portion of the URI is called the scheme-specific
   part (SSP), and can be quite general. (See, as one example, the URI
   scheme for SNMP [RFC4088]). Note that scheme-specific syntactical
   and semantic restrictions may be more constraining than the basic
   rules of RFC 3986. Section 3.1 of RFC 3986 provides guidance on the
   syntax of scheme names.

   URI schemes are a key concept in the DTN architecture, and evolved
   from an earlier concept called regions, which were tied more closely
   to assumptions of the network topology. Using URIs, significant
   flexibility is attained in the structuring of EIDs. They might, for
   example, be constructed based on DNS names, or might look like

"expressions of interest" or forms of database-like queries as in a
directed diffusion-routed network [IGE00] or in intentional naming
[WSBL99]. As names, EIDs are not required to be related to routing
or topological organization. Such a relationship is not prohibited,
however, and in some environments using EIDs this way may be
advantageous.

A single EID may refer to an endpoint containing more than one DTN
node, as suggested above. It is the responsibility of a scheme
designer to define how to interpret the SSP of an EID so as to
determine whether it refers to a unicast, multicast, or anycast set
of nodes. See Section 3.4 for more details.

URIs are constructed based on rules specified in RFC 3986, using the
US-ASCII character set. However, note this excerpt from RFC 3986,
Section 1.2.1, on dealing with characters that cannot be represented
by US-ASCII: "Percent-encoded octets (Section 2.1) may be used
within a URI to represent characters outside the range of the US-
ASCII coded character set if this representation is allowed by the
scheme or by the protocol element in which the URI is referenced.
Such a definition should specify the character encoding used to map
those characters to octets prior to being percent-encoded for the
URI".

3.3.2. Late Binding

Binding means interpreting the SSP of an EID for the purpose of
carrying an associated message towards a destination. For example,
binding might require mapping an EID to a next-hop EID or to a lower-
layer address for transmission. "Late binding" means that the
binding of a bundle's destination to a particular set of destination
identifiers or addresses does not necessarily happen at the bundle
source. Because the destination EID is potentially re-interpreted at
each hop, the binding may occur at the source, during transit, or
possibly at the destination(s). This contrasts with the name-to-
address binding of Internet communications where a DNS lookup at the
source fixes the IP address of the destination node before data is
sent. Such a circumstance would be considered "early binding"
because the name-to-address translation is performed prior to data
being sent into the network.

In a frequently-disconnected network, late binding may be
advantageous because the transit time of a message may exceed the
validity time of a binding, making binding at the source impossible
or invalid. Furthermore, use of name-based routing with late binding
may reduce the amount of administrative (mapping) information that

must propagate through the network, and may also limit the scope of
mapping synchronization requirements to a local topological
neighborhood where changes are made.

3.4. Anycast and Multicast

As mentioned above, an EID may refer to an endpoint containing one or
more DTN nodes. When referring to a group of size greater than one,
the delivery semantics may be of either the anycast or multicast
variety (broadcast is considered to be of the multicast variety).
For anycast group delivery, a bundle is delivered to one node among a
group of potentially many nodes, and for multicast delivery it is
intended to be delivered to all of them, subject to the normal DTN
class of service and maximum useful lifetime semantics.

Multicast group delivery in a DTN presents an unfamiliar issue with
respect to group membership. In relatively low-delay networks, such
as the Internet, nodes may be considered to be part of the group if
they have expressed interest to join it "recently". In a DTN,
however, nodes may wish to receive data sent to a group during an
interval of time earlier than when they are actually able to receive
it [ZAZ05]. More precisely, an application expresses its desire to
receive data sent to EID e at time t. Prior to this, during the
interval [t0, t1], t > t1, data may have been generated for group e.
For the application to receive any of this data, the data must be
available a potentially long time after senders have ceased sending
to the group. Thus, the data may need to be stored within the
network in order to support temporal group semantics of this kind.
How to design and implement this remains a research issue, as it is
likely to be at least as hard as problems related to reliable
multicast.

3.5. Priority Classes

The DTN architecture offers *relative* measures of priority (low,
medium, high) for delivering ADUs. These priorities differentiate
traffic based upon an application's desire to affect the delivery
urgency for ADUs, and are carried in bundle blocks generated by the
bundle layer based on information specified by the application.

The (U.S. or similar) Postal Service provides a strong metaphor for
the priority classes offered by the forwarding abstraction offered by
the DTN architecture. Traffic is generally not interactive and is
often one-way. There are generally no strong guarantees of timely
delivery, yet there are some forms of class of service, reliability,
and security.

We have defined three relative priority classes to date. These
priority classes typically imply some relative scheduling
prioritization among bundles in queue at a sender:

- Bulk - Bulk bundles are shipped on a "least effort" basis. No
  bundles of this class will be shipped until all bundles of other
  classes bound for the same destination and originating from the
  same source have been shipped.

- Normal - Normal-class bundles are shipped prior to any bulk-class
  bundles and are otherwise the same as bulk bundles.

- Expedited - Expedited bundles, in general, are shipped prior to
  bundles of other classes and are otherwise the same.

Applications specify their requested priority class and data lifetime
(see below) for each ADU they send. This information, coupled with
policy applied at DTN nodes that select how messages are forwarded
and which routing algorithms are in use, affects the overall
likelihood and timeliness of ADU delivery.

The priority class of a bundle is only required to relate to other
bundles from the same source. This means that a high priority bundle
from one source may not be delivered faster (or with some other
superior quality of service) than a medium priority bundle from a
different source. It does mean that a high priority bundle from one
source will be handled preferentially to a lower priority bundle sent
from the same source.

Depending on a particular DTN node's forwarding/scheduling policy,
priority may or may not be enforced across different sources. That
is, in some DTN nodes, expedited bundles might always be sent prior
to any bulk bundles, irrespective of source. Many variations are
possible.

3.6. Postal-Style Delivery Options and Administrative Records

Continuing with the postal analogy, the DTN architecture supports
several delivery options that may be selected by an application when
it requests the transmission of an ADU. In addition, the
architecture defines two types of administrative records: "status
reports" and "signals". These records are bundles that provide
information about the delivery of other bundles, and are used in
conjunction with the delivery options.

3.6.1. Delivery Options

   We have defined eight delivery options. Applications sending an ADU
   (the "subject ADU") may request any combination of the following,
   which are carried in each of the bundles produced ("sent bundles") by
   the bundle layer resulting from the application's request to send the
   subject ADU:

   - Custody Transfer Requested - requests sent bundles be delivered
     with enhanced reliability using custody transfer procedures. Sent
     bundles will be transmitted by the bundle layer using reliable
     transfer protocols (if available), and the responsibility for
     reliable delivery of the bundle to its destination(s) may move
     among one or more "custodians" in the network. This capability is
     described in more detail in Section 3.10.

   - Source Node Custody Acceptance Required - requires the source DTN
     node to provide custody transfer for the sent bundles. If custody
     transfer is not available at the source when this delivery option
     is requested, the requested transmission fails. This provides a
     means for applications to insist that the source DTN node take
     custody of the sent bundles (e.g., by storing them in persistent
     storage).

   - Report When Bundle Delivered - requests a (single) Bundle Delivery
     Status Report be generated when the subject ADU is delivered to its
     intended recipient(s). This request is also known as "return-
     receipt".

   - Report When Bundle Acknowledged by Application - requests an
     Acknowledgement Status Report be generated when the subject ADU is
     acknowledged by a receiving application. This only happens by
     action of the receiving application, and differs from the Bundle
     Delivery Status Report. It is intended for cases where the
     application may be acting as a form of application layer gateway
     and wishes to indicate the status of a protocol operation external
     to DTN back to the requesting source. See Section 11 for more
     details.

   - Report When Bundle Received - requests a Bundle Reception Status
     Report be generated when each sent bundle arrives at a DTN node.
     This is designed primarily for diagnostic purposes.

   - Report When Bundle Custody Accepted - requests a Custody
     Acceptance Status Report be generated when each sent bundle has
     been accepted using custody transfer. This is designed primarily
     for diagnostic purposes.

    - Report When Bundle Forwarded - requests a Bundle Forwarding Status
      Report be generated when each sent bundle departs a DTN node after
      forwarding. This is designed primarily for diagnostic purposes.

    - Report When Bundle Deleted - requests a Bundle Deletion Status
      Report be generated when each sent bundle is deleted at a DTN node.
      This is designed primarily for diagnostic purposes.

   The first four delivery options are designed for ordinary use by
   applications. The last four are designed primarily for diagnostic
   purposes and their use may be restricted or limited in environments
   subject to congestion or attack.

   If the security procedures defined in [DTNSEC] are also enabled, then
   three additional delivery options become available:

   - Confidentiality Required - requires the subject ADU be made secret
     from parties other than the source and the members of the
     destination EID.

   - Authentication Required - requires all non-mutable fields in the
     bundle blocks of the sent bundles (i.e., those which do not change
     as the bundle is forwarded) be made strongly verifiable (i.e.,
     cryptographically strong). This protects several fields, including
     the source and destination EIDs and the bundle's data. See Section
     3.7 and [BSPEC] for more details.

   - Error Detection Required - requires modifications to the non-
     mutable fields of each sent bundle be made detectable with high
     probability at each destination.

3.6.2. Administrative Records: Bundle Status Reports and Custody
        Signals

   Administrative records are used to report status information or error
   conditions related to the bundle layer. There are two types of
   administrative records defined: bundle status reports (BSRs) and
   custody signals. Administrative records correspond (approximately)
   to messages in the ICMP protocol in IP [RFC792]. In ICMP, however,
   messages are returned to the source. In DTN, they are instead
   directed to the report-to EID for BSRs and the EID of the current
   custodian for custody signals, which might differ from the source's
   EID. Administrative records are sent as bundles with a source EID
   set to one of the EIDs associated with the DTN node generating the
   administrative record. In some cases, arrival of a single bundle or
   bundle fragment may elicit multiple administrative records (e.g., in
   the case where a bundle is replicated for multicast forwarding).

The following BSRs are currently defined (also see [BSPEC] for more
details):

- Bundle Reception - sent when a bundle arrives at a DTN node.
  Generation of this message may be limited by local policy.

- Custody Acceptance - sent when a node has accepted custody of a
  bundle with the Custody Transfer Requested option set. Generation
  of this message may be limited by local policy.

- Bundle Forwarded - sent when a bundle containing a Report When
  Bundle Forwarded option departs from a DTN node after having been
  forwarded. Generation of this message may be limited by local
  policy.

- Bundle Deletion - sent from a DTN node when a bundle containing a
  Report When Bundle Deleted option is discarded. This can happen
  for several reasons, such as expiration. Generation of this
  message may be limited by local policy but is required in cases
  where the deletion is performed by a bundle's current custodian.

- Bundle Delivery - sent from a final recipient's (destination) node
  when a complete ADU comprising sent bundles containing Report When
  Bundle Delivered options is consumed by an application.

- Acknowledged by application - sent from a final recipient's
  (destination) node when a complete ADU comprising sent bundles
  containing Application Acknowledgment options has been processed by
  an application. This generally involves specific action on the
  receiving application's part.

In addition to the status reports, the custody signal is currently
defined to indicate the status of a custody transfer. These are sent
to the current-custodian EID contained in an arriving bundle:

- Custody Signal - indicates that custody has been successfully
  transferred. This signal appears as a Boolean indicator, and may
  therefore indicate either a successful or a failed custody transfer
  attempt.

Administrative records must reference a received bundle. This is
accomplished by a method for uniquely identifying bundles based on a
transmission timestamp and sequence number discussed in Section 3.12.

3.7. Primary Bundle Fields

   The bundles carried between and among DTN nodes obey a standard
   bundle protocol specified in [BSPEC]. Here we provide an overview of
   most of the fields carried with every bundle. The protocol is
   designed with a mandatory primary block, an optional payload block
   (which contains the ADU data itself), and a set of optional extension
   blocks. Blocks may be cascaded in a way similar to extension headers
   in IPv6. The following selected fields are all present in the
   primary block, and therefore are present for every bundle and
   fragment:

   - Creation Timestamp - a concatenation of the bundle's creation time
     and a monotonically increasing sequence number such that the
     creation timestamp is guaranteed to be unique for each ADU
     originating from the same source. The creation timestamp is based
     on the time-of-day an application requested an ADU to be sent (not
     when the corresponding bundle(s) are sent into the network). DTN
     nodes are assumed to have a basic time synchronization capability
     (see Section 3.12).

   - Lifespan - the time-of-day at which the message is no longer
     useful. If a bundle is stored in the network (including the
     source's DTN node) when its lifespan is reached, it may be
     discarded. The lifespan of a bundle is expressed as an offset
     relative to its creation time.

   - Class of Service Flags - indicates the delivery options and
     priority class for the bundle. Priority classes may be one of
     bulk, normal, or expedited. See Section 3.6.1.

   - Source EID - EID of the source (the first sender).

   - Destination EID - EID of the destination (the final intended
     recipient(s)).

   - Report-To Endpoint ID - an EID identifying where reports (return-
     receipt, route-tracing functions) should be sent. This may or may
     not identify the same endpoint as the Source EID.

   - Custodian EID - EID of the current custodian of a bundle (if any).

   The payload block indicates information about the contained payload
   (e.g., its length) and the payload itself. In addition to the fields
   found in the primary and payload blocks, each bundle may have fields
   in additional blocks carried with each bundle. See [BSPEC] for
   additional details.

3.8. Routing and Forwarding

   The DTN architecture provides a framework for routing and forwarding
   at the bundle layer for unicast, anycast, and multicast messages.
   Because nodes in a DTN network might be interconnected using more
   than one type of underlying network technology, a DTN network is best
   described abstractly using a *multigraph* (a graph where vertices may
   be interconnected with more than one edge). Edges in this graph are,
   in general, time-varying with respect to their delay and capacity and
   directional because of the possibility of one-way connectivity. When
   an edge has zero capacity, it is considered to not be connected.

   Because edges in a DTN graph may have significant delay, it is
   important to distinguish where time is measured when expressing an
   edge's capacity or delay. We adopt the convention of expressing
   capacity and delay as functions of time where time is measured at the
   point where data is inserted into a network edge. For example,
   consider an edge having capacity $C(t)$ and delay $D(t)$ at time t. If B
   bits are placed in this edge at time t, they completely arrive by
   time $t + D(t) + (1/C(t))*B$. We assume $C(t)$ and $D(t)$ do not change
   significantly during the interval $[t, t+D(t)+(1/C(t))*B]$.

   Because edges may vary between positive and zero capacity, it is
   possible to describe a period of time (interval) during which the
   capacity is strictly positive, and the delay and capacity can be
   considered to be constant [AF03]. This period of time is called a
   "contact". In addition, the product of the capacity and the interval
   is known as a contact's "volume". If contacts and their volumes are
   known ahead of time, intelligent routing and forwarding decisions can
   be made (optimally for small networks) [JFP04]. Optimally using a
   contact's volume, however, requires the ability to divide large ADUs
   and bundles into smaller routable units. This is provided by DTN
   fragmentation (see Section 3.9).

   When delivery paths through a DTN graph are lossy or contact
   intervals and volumes are not known precisely ahead of time, routing
   computations become especially challenging. How to handle these
   situations is an active area of work in the (emerging) research area
   of delay tolerant networking.

3.8.1. Types of Contacts

   Contacts typically fall into one of several categories, based largely
   on the predictability of their performance characteristics and
   whether some action is required to bring them into existence. To
   date, the following major types of contacts have been defined:

   Persistent Contacts

      Persistent contacts are always available (i.e., no connection-
      initiation action is required to instantiate a persistent
      contact). An 'always-on' Internet connection such as a DSL or
      Cable Modem connection would be a representative of this class.

   On-Demand Contacts

      On-Demand contacts require some action in order to instantiate,
      but then function as persistent contacts until terminated. A
      dial-up connection is an example of an On-Demand contact (at
      least, from the viewpoint of the dialer; it may be viewed as an
      Opportunistic Contact, below, from the viewpoint of the dial-up
      service provider).

   Intermittent - Scheduled Contacts

      A scheduled contact is an agreement to establish a contact at a
      particular time, for a particular duration. An example of a
      scheduled contact is a link with a low-earth orbiting satellite.
      A node's list of contacts with the satellite can be constructed
      from the satellite's schedule of view times, capacities, and
      latencies. Note that for networks with substantial delays, the
      notion of the "particular time" is delay-dependent. For example,
      a single scheduled contact between Earth and Mars would not be at
      the same instant in each location, but would instead be offset by
      the (non-negligible) propagation delay.

   Intermittent - Opportunistic Contacts

      Opportunistic contacts are not scheduled, but rather present
      themselves unexpectedly. For example, an unscheduled aircraft
      flying overhead and beaconing, advertising its availability for
      communication, would present an opportunistic contact. Another
      type of opportunistic contact might be via an infrared or
      Bluetooth communication link between a personal digital assistant
      (PDA) and a kiosk in an airport concourse. The opportunistic
      contact begins as the PDA is brought near the kiosk, lasting an
      undetermined amount of time (i.e., until the link is lost or
      terminated).

   Intermittent - Predicted Contacts

      Predicted contacts are based on no fixed schedule, but rather are
      predictions of likely contact times and durations based on a
      history of previously observed contacts or some other information.
      Given a great enough confidence in a predicted contact, routes may

be chosen based on this information. This is an active research
area, and a few approaches having been proposed [LFC05].

3.9. Fragmentation and Reassembly

   DTN fragmentation and reassembly are designed to improve the
   efficiency of bundle transfers by ensuring that contact volumes are
   fully utilized and by avoiding retransmission of partially-forwarded
   bundles. There are two forms of DTN fragmentation/reassembly:

   Proactive Fragmentation

      A DTN node may divide a block of application data into multiple
      smaller blocks and transmit each such block as an independent
      bundle. In this case, the *final destination(s)* are responsible
      for extracting the smaller blocks from incoming bundles and
      reassembling them into the original larger bundle and, ultimately,
      ADU. This approach is called proactive fragmentation because it
      is used primarily when contact volumes are known (or predicted) in
      advance.

   Reactive Fragmentation

      DTN nodes sharing an edge in the DTN graph may fragment a bundle
      cooperatively when a bundle is only partially transferred. In
      this case, the receiving bundle layer modifies the incoming bundle
      to indicate it is a fragment, and forwards it normally. The
      previous- hop sender may learn (via convergence-layer protocols,
      see Section 6) that only a portion of the bundle was delivered to
      the next hop, and send the remaining portion(s) when subsequent
      contacts become available (possibly to different next-hops if
      routing changes). This is called reactive fragmentation because
      the fragmentation process occurs after an attempted transmission
      has taken place.

      As an example, consider a ground station G, and two store-and-
      forward satellites S1 and S2, in opposite low-earth orbit. While
      G is transmitting a large bundle to S1, a reliable transport layer
      protocol below the bundle layer at each indicates the transmission
      has terminated, but that half the transfer has completed
      successfully. In this case, G can form a smaller bundle fragment
      consisting of the second half of the original bundle and forward
      it to S2 when available. In addition, S1 (now out of range of G)
      can form a new bundle consisting of the first half of the original
      bundle and forward it to whatever next hop(s) it deems
      appropriate.

The reactive fragmentation capability is not required to be available
in every DTN implementation, as it requires a certain level of
support from underlying protocols that may not be present, and
presents significant challenges with respect to handling digital
signatures and authentication codes on messages. When a signed
message is only partially received, most message authentication codes
will fail. When DTN security is present and enabled, it may
therefore be necessary to proactively fragment large bundles into
smaller units that are more convenient for digital signatures.

Even if reactive fragmentation is not present in an implementation,
the ability to reassemble fragments at a destination is required in
order to support DTN fragmentation. Furthermore, for contacts with
volumes that are small compared to typical bundle sizes, some
incremental delivery approach must be used (e.g., checkpoint/restart)
to prevent data delivery livelock. Reactive fragmentation is one
such approach, but other protocol layers could potentially handle
this issue as well.

3.10. Reliability and Custody Transfer

The most basic service provided by the bundle layer is
unacknowledged, prioritized (but not guaranteed) unicast message
delivery. It also provides two options for enhancing delivery
reliability: end-to-end acknowledgments and custody transfer.
Applications wishing to implement their own end-to-end message
reliability mechanisms are free to utilize the acknowledgment. The
custody transfer feature of the DTN architecture only specifies a
coarse-grained retransmission capability, described next.

Transmission of bundles with the Custody Transfer Requested option
specified generally involves moving the responsibility for reliable
delivery of an ADU's bundles among different DTN nodes in the
network. For unicast delivery, this will typically involve moving
bundles "closer" (in terms of some routing metric) to their ultimate
destination(s), and retransmitting when necessary. The nodes
receiving these bundles along the way (and agreeing to accept the
reliable delivery responsibility) are called "custodians". The
movement of a bundle (and its delivery responsibility) from one node
to another is called a "custody transfer". It is analogous to a
database commit transaction [FHM03]. The exact meaning and design of
custody transfer for multicast and anycast delivery remains to be
fully explored.

Custody transfer allows the source to delegate retransmission
responsibility and recover its retransmission-related resources
relatively soon after sending a bundle (on the order of the minimum
round-trip time to the first bundle hop(s)). Not all nodes in a DTN

are required by the DTN architecture to accept custody transfers, so
it is not a true 'hop-by-hop' mechanism. For example, some nodes may
have sufficient storage resources to sometimes act as custodians, but
may elect to not offer such services when congested or running low on
power.

The existence of custodians can alter the way DTN routing is
performed. In some circumstances, it may be beneficial to move a
bundle to a custodian as quickly as possible even if the custodian is
further away (in terms of distance, time or some routing metric) from
the bundle's final destination(s) than some other reachable node.
Designing a system with this capability involves constructing more
than one routing graph, and is an area of continued research.

Custody transfer in DTN not only provides a method for tracking
bundles that require special handling and identifying DTN nodes that
participate in custody transfer, it also provides a (weak) mechanism
for enhancing the reliability of message delivery. Generally
speaking, custody transfer relies on underlying reliable delivery
protocols of the networks that it operates over to provide the
primary means of reliable transfer from one bundle node to the next
(set). However, when custody transfer is requested, the bundle layer
provides an additional coarse-grained timeout and retransmission
mechanism and an accompanying (bundle-layer) custodian-to-custodian
acknowledgment signaling mechanism. When an application does *not*
request custody transfer, this bundle layer timeout and
retransmission mechanism is typically not employed, and successful
bundle layer delivery depends solely on the reliability mechanisms of
the underlying protocols.

When a node accepts custody for a bundle that contains the Custody
Transfer Requested option, a Custody Transfer Accepted Signal is sent
by the bundle layer to the Current Custodian EID contained in the
primary bundle block. In addition, the Current Custodian EID is
updated to contain one of the forwarding node's (unicast) EIDs before
the bundle is forwarded.

When an application requests an ADU to be delivered with custody
transfer, the request is advisory. In some circumstances, a source
of a bundle for which custody transfer has been requested may not be
able to provide this service. In such circumstances, the subject
bundle may traverse multiple DTN nodes before it obtains a custodian.
Bundles in this condition are specially marked with their Current
Custodian EID field set to a null endpoint. In cases where
applications wish to require the source to take custody of the
bundle, they may supply the Source Node Custody Acceptance Required

delivery option. This may be useful to applications that desire a
continuous "chain" of custody or that wish to exit after being
ensured their data is safely held in a custodian.

In a DTN network where one or more custodian-to-custodian hops are
strictly one directional (and cannot be reversed), the DTN custody
transfer mechanism will be affected over such hops due to the lack of
any way to receive a custody signal (or any other information) back
across the path, resulting in the expiration of the bundle at the
ingress to the one-way hop. This situation does not necessarily mean
the bundle has been lost; nodes on the other side of the hop may
continue to transfer custody, and the bundle may be delivered
successfully to its destination(s). However, in this circumstance a
source that has requested to receive expiration BSRs for this bundle
will receive an expiration report for the bundle, and possibly
conclude (incorrectly) that the bundle has been discarded and not
delivered. Although this problem cannot be fully solved in this
situation, a mechanism is provided to help ameliorate the seemingly
incorrect information that may be reported when the bundle expires
after having been transferred over a one-way hop. This is
accomplished by the node at the ingress to the one-way hop reporting
the existence of a known one-way path using a variant of a bundle
status report. These types of reports are provided if the subject
bundle requests the report using the 'Report When Bundle Forwarded'
delivery option.

3.11. DTN Support for Proxies and Application Layer Gateways

One of the aims of DTN is to provide a common method for
interconnecting application layer gateways and proxies. In cases
where existing Internet applications can be made to tolerate delays,
local proxies can be constructed to benefit from the existing
communication capabilities provided by DTN [S05, T02]. Making such
proxies compatible with DTN reduces the burden on the proxy author
from being concerned with how to implement routing and reliability
management and allows existing TCP/IP-based applications to operate
unmodified over a DTN-based network.

When DTN is used to provide a form of tunnel encapsulation for other
protocols, it can be used in constructing overlay networks comprised
of application layer gateways. The application acknowledgment
capability is designed for such circumstances. This provides a
common way for remote application layer gateways to signal the
success or failure of non-DTN protocol operations initiated as a
result of receiving DTN ADUs. Without this capability, such
indicators would have to be implemented by applications themselves in
non-standard ways.

3.12. Timestamps and Time Synchronization

   The DTN architecture depends on time synchronization among DTN nodes
   (supported by external, non-DTN protocols) for four primary purposes:
   bundle and fragment identification, routing with scheduled or
   predicted contacts, bundle expiration time computations, and
   application registration expiration.

   Bundle identification and expiration are supported by placing a
   creation timestamp and an explicit expiration field (expressed in
   seconds after the source timestamp) in each bundle. The origination
   timestamps on arriving bundles are made available to consuming
   applications in ADUs they receive by some system interface function.
   Each set of bundles corresponding to an ADU is required to contain a
   timestamp unique to the sender's EID. The EID, timestamp, and data
   offset/length information together uniquely identify a bundle.
   Unique bundle identification is used for a number of purposes,
   including custody transfer and reassembly of bundle fragments.

   Time is also used in conjunction with application registrations.
   When an application expresses its desire to receive ADUs destined for
   a particular EID, this registration is only maintained for a finite
   period of time, and may be specified by the application. For
   multicast registrations, an application may also specify a time range
   or "interest interval" for its registration. In this case, traffic
   sent to the specified EID any time during the specified interval will
   eventually be delivered to the application (unless such traffic has
   expired due to the expiration time provided by the application at the
   source or some other reason prevents such delivery).

3.13. Congestion and Flow Control at the Bundle Layer

   The subject of congestion control and flow control at the bundle
   layer is one on which the authors of this document have not yet
   reached complete consensus. We have unresolved concerns about the
   efficiency and efficacy of congestion and flow control schemes
   implemented across long and/or highly variable delay environments,
   especially with the custody transfer mechanism that may require nodes
   to retain bundles for long periods of time.

   For the purposes of this document, we define "flow control" as a
   means of assuring that the average rate at which a sending node
   transmits data to a receiving node does not exceed the average rate
   at which the receiving node is prepared to receive data from that
   sender. (Note that this is a generalized notion of flow control,
   rather than one that applies only to end-to-end communication.) We
   define "congestion control" as a means of assuring that the aggregate
   rate at which all traffic sources inject data into a network does not

exceed the maximum aggregate rate at which the network can deliver
data to destination nodes over time. If flow control is propagated
backward from congested nodes toward traffic sources, then the flow
control mechanism can be used as at least a partial solution to the
problem of congestion as well.

DTN flow control decisions must be made within the bundle layer
itself based on information about resources (in this case, primarily
persistent storage) available within the bundle node. When storage
resources become scarce, a DTN node has only a certain degree of
freedom in handling the situation. It can always discard bundles
which have expired -- an activity DTN nodes should perform regularly
in any case. If it ordinarily is willing to accept custody for
bundles, it can cease doing so. If storage resources are available
elsewhere in the network, it may be able to make use of them in some
way for bundle storage. It can also discard bundles which have not
expired but for which it has not accepted custody. A node must avoid
discarding bundles for which it has accepted custody, and do so only
as a last resort. Determining when a node should engage in or cease
to engage in custody transfers is a resource allocation and
scheduling problem of current research interest.

In addition to the bundle layer mechanisms described above, a DTN
node may be able to avail itself of support from lower-layer
protocols in affecting its own resource utilization. For example, a
DTN node receiving a bundle using TCP/IP might intentionally slow
down its receiving rate by performing read operations less frequently
in order to reduce its offered load. This is possible because TCP
provides its own flow control, so reducing the application data
consumption rate could effectively implement a form of hop-by-hop
flow control. Unfortunately, it may also lead to head-of-line
blocking issues, depending on the nature of bundle multiplexing
within a TCP connection. A protocol with more relaxed ordering
constraints (e.g. SCTP [RFC2960]) might be preferable in such
circumstances.

Congestion control is an ongoing research topic.

3.14. Security

The possibility of severe resource scarcity in some delay-tolerant
networks dictates that some form of authentication and access control
to the network itself is required in many circumstances. It is not
acceptable for an unauthorized user to flood the network with traffic
easily, possibly denying service to authorized users. In many cases
it is also not acceptable for unauthorized traffic to be forwarded
over certain network links at all. This is especially true for
exotic, mission-critical links. In light of these considerations,

several goals are established for the security component of the DTN
architecture:

- Promptly prevent unauthorized applications from having their data
  carried through or stored in the DTN.

- Prevent unauthorized applications from asserting control over the
  DTN infrastructure.

- Prevent otherwise authorized applications from sending bundles at a
  rate or class of service for which they lack permission.

- Promptly discard bundles that are damaged or improperly modified in
  transit.

- Promptly detect and de-authorize compromised entities.

Many existing authentication and access control protocols designed
for operation in low-delay, connected environments may not perform
well in DTNs. In particular, updating access control lists and
revoking ("blacklisting") credentials may be especially difficult.
Also, approaches that require frequent access to centralized servers
to complete an authentication or authorization transaction are not
attractive. The consequences of these difficulties include delays in
the onset of communication, delays in detecting and recovering from
system compromise, and delays in completing transactions due to
inappropriate access control or authentication settings.

To help satisfy these security requirements in light of the
challenges, the DTN architecture adopts a standard but optionally
deployed security architecture [DTNSEC] that utilizes hop-by-hop and
end-to-end authentication and integrity mechanisms. The purpose of
using both approaches is to be able to handle access control for data
forwarding and storage separately from application-layer data
integrity. While the end-to-end mechanism provides authentication
for a principal such as a user (of which there may be many), the hop-
by-hop mechanism is intended to authenticate DTN nodes as legitimate
transceivers of bundles to each-other. Note that it is conceivable
to construct a DTN in which only a subset of the nodes participate in
the security mechanisms, resulting in a secure DTN overlay existing
atop an insecure DTN overlay. This idea is relatively new and is
still being explored.

In accordance with the goals listed above, DTN nodes discard traffic
as early as possible if authentication or access control checks fail.
This approach meets the goals of removing unwanted traffic from being
forwarded over specific high-value links, but also has the associated
benefit of making denial-of-service attacks considerably harder to

mount more generally, as compared with conventional Internet routers.
However, the obvious cost for this capability is potentially larger
computation and credential storage overhead required at DTN nodes.

For more detailed information on DTN security provisions, refer to
[DTNSEC] and [DTNSOV].

4. State Management Considerations

An important aspect of any networking architecture is its management
of state. This section describes the state managed at the bundle
layer and discusses how it is established and removed.

4.1. Application Registration State

In long/variable delay environments, an asynchronous application
interface seems most appropriate. Such interfaces typically include
methods for applications to register callback actions when certain
triggering events occur (e.g., when ADUs arrive). These
registrations create state information called application
registration state.

Application registration state is typically created by explicit
request of the application, and is removed by a separate explicit
request, but may also be removed by an application-specified timer
(it is thus "firm" state). In most cases, there must be a provision
for retaining this state across application and operating system
termination/restart conditions because a client/server bundle round-
trip time may exceed the requesting application's execution time (or
hosting system's uptime). In cases where applications are not
automatically restarted but application registration state remains
persistent, a method must be provided to indicate to the system what
action to perform when the triggering event occurs (e.g., restarting
some application, ignoring the event, etc.).

To initiate a registration and thereby establish application
registration state, an application specifies an Endpoint ID for which
it wishes to receive ADUs, along with an optional time value
indicating how long the registration should remain active. This
operation is somewhat analogous to the bind() operation in the common
sockets API.

For registrations to groups (i.e., joins), a time interval may also
be specified. The time interval refers to the range of origination
times of ADUs sent to the specified EID. See Section 3.4 above for
more details.

4.2. Custody Transfer State

   Custody transfer state includes information required to keep account
   of bundles for which a node has taken custody, as well as the
   protocol state related to transferring custody for one or more of
   them. The accounting-related state is created when a bundle is
   received. Custody transfer retransmission state is created when a
   transfer of custody is initiated by forwarding a bundle with the
   custody transfer requested delivery option specified. Retransmission
   state and accounting state may be released upon receipt of one or
   more Custody Transfer Succeeded signals, indicating custody has been
   moved. In addition, the bundle's expiration time (possibly mitigated
   by local policy) provides an upper bound on the time when this state
   is purged from the system in the event that it is not purged
   explicitly due to receipt of a signal.

4.3. Bundle Routing and Forwarding State

   As with the Internet architecture, we distinguish between routing and
   forwarding. Routing refers to the execution of a (possibly
   distributed) algorithm for computing routing paths according to some
   objective function (see [JFP04], for example). Forwarding refers to
   the act of moving a bundle from one DTN node to another. Routing
   makes use of routing state (the RIB, or routing information base),
   while forwarding makes use of state derived from routing, and is
   maintained as forwarding state (the FIB, or forwarding information
   base). The structure of the FIB and the rules for maintaining it are
   implementation choices. In some DTNs, exchange of information used
   to update state in the RIB may take place on network paths distinct
   from those where exchange of application data takes place.

   The maintenance of state in the RIB is dependent on the type of
   routing algorithm being used. A routing algorithm may consider
   requested class of service and the location of potential custodians
   (for custody transfer, see section 3.10), and this information will
   tend to increase the size of the RIB. The separation between FIB and
   RIB is not required by this document, as these are implementation
   details to be decided by system implementers. The choice of routing
   algorithms is still under study.

   Bundles may occupy queues in nodes for a considerable amount of time.
   For unicast or anycast delivery, the amount of time is likely to be
   the interval between when a bundle arrives at a node and when it can
   be forwarded to its next hop. For multicast delivery of bundles,
   this could be significantly longer, up to a bundle's expiration time.
   This situation occurs when multicast delivery is utilized in such a
   way that nodes joining a group can obtain information previously sent
   to the group. In such cases, some nodes may act as "archivers" that

provide copies of bundles to new participants that have already been
delivered to other participants.

4.4. Security-Related State

The DTN security approach described in [DTNSEC], when used, requires
maintenance of state in all DTN nodes that use it. All such nodes
are required to store their own private information (including their
own policy and authentication material) and a block of information
used to verify credentials. Furthermore, in most cases, DTN nodes
will cache some public information (and possibly the credentials) of
their next-hop (bundle) neighbors. All cached information has
expiration times, and nodes are responsible for acquiring and
distributing updates of public information and credentials prior to
the expiration of the old set (in order to avoid a disruption in
network service).

In addition to basic end-to-end and hop-by-hop authentication, access
control may be used in a DTN by one or more mechanisms such as
capabilities or access control lists (ACLs). ACLs would represent
another block of state present in any node that wishes to enforce
security policy. ACLs are typically initialized at node
configuration time and may be updated dynamically by DTN bundles or
by some out of band technique. Capabilities or credentials may be
revoked, requiring the maintenance of a revocation list ("black
list", another form of state) to check for invalid authentication
material that has already been distributed.

Some DTNs may implement security boundaries enforced by selected
nodes in the network, where end-to-end credentials may be checked in
addition to checking the hop-by-hop credentials. (Doing so may
require routing to be adjusted to ensure all bundles comprising each
ADU pass through these points.) Public information used to verify
end-to-end authentication will typically be cached at these points.

4.5. Policy and Configuration State

DTN nodes will contain some amount of configuration and policy
information. Such information may alter the behavior of bundle
forwarding. Examples of policy state include the types of
cryptographic algorithms and access control procedures to use if DTN
security is employed, whether nodes may become custodians, what types
of convergence layer (see Section 6) and routing protocols are in
use, how bundles of differing priorities should be scheduled, where
and for how long bundles and other data is stored, what status
reports may be generated or at what rate, etc.

5. Application Structuring Issues

   DTN bundle delivery is intended to operate in a delay-tolerant
   fashion over a broad range of network types. This does not mean
   there *must* be large delays in the network; it means there *may* be
   very significant delays (including extended periods of disconnection
   between sender and intended recipient(s)). The DTN protocols are
   delay tolerant, so applications using them must also be delay
   tolerant in order to operate effectively in environments subject to
   significant delay or disruption.

   The communication primitives provided by the DTN architecture are
   based on asynchronous, message-oriented communication which differs
   from conversational request/response communication. In general,
   applications should attempt to include enough information in an ADU
   so that it may be treated as an independent unit of work by the
   network and receiver(s). The goal is to minimize synchronous
   interchanges between applications that are separated by a network
   characterized by long and possibly highly variable delays. A single
   file transfer request message, for example, might include
   authentication information, file location information, and requested
   file operation (thus "bundling" this information together).
   Comparing this style of operation to a classic FTP transfer, one sees
   that the bundled model can complete in one round trip, whereas an FTP
   file "put" operation can take as many as eight round trips to get to
   a point where file data can flow [DFS02].

   Delay-tolerant applications must consider additional factors beyond
   the conversational implications of long delay paths. For example, an
   application may terminate (voluntarily or not) between the time it
   sends a message and the time it expects a response. If this
   possibility has been anticipated, the application can be "re-
   instantiated" with state information saved in persistent storage.
   This is an implementation issue, but also an application design
   consideration.

   Some consideration of delay-tolerant application design can result in
   applications that work reasonably well in low-delay environments, and
   that do not suffer extraordinarily in high or highly-variable delay
   environments.

6. Convergence Layer Considerations for Use of Underlying Protocols

   Implementation experience with the DTN architecture has revealed an
   important architectural construct and interface for DTN nodes
   [DBFJHP04]. Not all underlying protocols in different protocol
   families provide the same exact functionality, so some additional
   adaptation or augmentation on a per-protocol or per-protocol-family

basis may be required. This adaptation is accomplished by a set of
convergence layers placed between the bundle layer and underlying
protocols. The convergence layers manage the protocol-specific
details of interfacing with particular underlying protocols and
present a consistent interface to the bundle layer.

The complexity of one convergence layer may vary substantially from
another, depending on the type of underlying protocol it adapts. For
example, a TCP/IP convergence layer for use in the Internet might
only have to add message boundaries to TCP streams, whereas a
convergence layer for some network where no reliable transport
protocol exists might be considerably more complex (e.g., it might
have to implement reliability, fragmentation, flow-control, etc.) if
reliable delivery is to be offered to the bundle layer.

As convergence layers implement protocols above and beyond the basic
bundle protocol specified in [BSPEC], they will be defined in their
own documents (in a fashion similar to the way encapsulations for IP
datagrams are specified on a per-underlying-protocol basis, such as
in RFC 894 [RFC894]).

7. Summary

The DTN architecture addresses many of the problems of heterogeneous
networks that must operate in environments subject to long delays and
discontinuous end-to-end connectivity. It is based on asynchronous
messaging and uses postal mail as a model of service classes and
delivery semantics. It accommodates many different forms of
connectivity, including scheduled, predicted, and opportunistically
connected delivery paths. It introduces a novel approach to end-to-
end reliability across frequently partitioned and unreliable
networks. It also proposes a model for securing the network
infrastructure against unauthorized access.

It is our belief that this architecture is applicable to many
different types of challenged environments.

8. Security Considerations

Security is an integral concern for the design of the Delay Tolerant
Network Architecture, but its use is optional. Sections 3.6.1, 3.14,
and 4.4 of this document present some factors to consider for
securing the DTN architecture, but separate documents [DTNSOV] and
[DTNSEC] define the security architecture in much more detail.

9. IANA Considerations

   This document specifies the architecture for Delay Tolerant
   Networking, which uses Internet-standard URIs for its Endpoint
   Identifiers. URIs intended for use with DTN should be compliant with
   the guidelines given in [RFC3986].

10. Normative References

   [RFC3986]   Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform
               Resource Identifier (URI): Generic Syntax", STD 66, RFC
               3986, January 2005.

11. Informative References

   [IPN01]   InterPlaNetary Internet Project, Internet Society IPN
               Special Interest Group, http://www.ipnsig.org.

   [SB03]   S. Burleigh, et al., "Delay-Tolerant Networking - An
               Approach to Interplanetary Internet", IEEE Communications
               Magazine, July 2003.

   [FW03]   F. Warthman, "Delay-Tolerant Networks (DTNs): A Tutorial
               v1.1", Wartham Associates, 2003. Available from
               http://www.dtnrg.org.

   [KF03]   K. Fall, "A Delay-Tolerant Network Architecture for
               Challenged Internets", Proceedings SIGCOMM, Aug 2003.

   [JFP04]   S. Jain, K. Fall, R. Patra, "Routing in a Delay Tolerant
               Network", Proceedings SIGCOMM, Aug/Sep 2004.

   [DFS02]   R. Durst, P. Feighery, K. Scott, "Why not use the
               Standard Internet Suite for the Interplanetary
               Internet?", MITRE White Paper, 2002. Available from
               http://www.ipnsig.org/reports/TCP_IP.pdf.

   [CK74]   V. Cerf, R. Kahn, "A Protocol for Packet Network
               Intercommunication", IEEE Trans. on Comm., COM-22(5), May
               1974.

   [IGE00]   C. Intanagonwiwat, R. Govindan, D. Estrin, "Directed
               Diffusion: A Scalable and Robust Communication Paradigm
               for Sensor Networks", Proceedings MobiCOM, Aug 2000.

   [WSBL99]   W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, J. Lilley,
                 "The Design and Implementation of an Intentional Naming
                 System", Proc. 17th ACM SOSP, Kiawah Island, SC, Dec.
                 1999.

   [CT90]   D. Clark, D. Tennenhouse, "Architectural Considerations
                 for a New Generation of Protocols", Proceedings SIGCOMM,
                 1990.

   [ISCHEMES]   IANA, Uniform Resource Identifer (URI) Schemes,
                 http://www.iana.org/assignments/uri-schemes.html.

   [JDPF05]   S. Jain, M. Demmer, R. Patra, K. Fall, "Using Redundancy
                 to Cope with Failures in a Delay Tolerant Network",
                 Proceedings SIGCOMM, 2005.

   [WJMF05]   Y. Wang, S. Jain, M. Martonosi, K. Fall, "Erasure Coding
                 Based Routing in Opportunistic Networks", Proceedings
                 SIGCOMM Workshop on Delay Tolerant Networks, 2005.

   [ZAZ05]   W. Zhao, M. Ammar, E. Zegura, "Multicast in Delay
                 Tolerant Networks", Proceedings SIGCOMM Workshop on Delay
                 Tolerant Networks, 2005.

   [LFC05]   J. Leguay, T. Friedman, V. Conan, "DTN Routing in a
                 Mobility Pattern Space", Proceedings SIGCOMM Workshop on
                 Delay Tolerant Networks, 2005.

   [AF03]   J. Alonso, K. Fall, "A Linear Programming Formulation of
                 Flows over Time with Piecewise Constant Capacity and
                 Transit Times", Intel Research Technical Report IRB-TR-
                 03-007, June 2003.

   [FHM03]   K. Fall, W. Hong, S. Madden, "Custody Transfer for
                 Reliable Delivery in Delay Tolerant Networks", Intel
                 Research Technical Report IRB-TR-03-030, July 2003.

   [BSPEC]   K. Scott, S. Burleigh, "Bundle Protocol Specification",
                 Work in Progress, December 2006.

   [DTNSEC]   S. Symington, S. Farrell, H. Weiss, "Bundle Security
                 Protocol Specification", Work in Progress, October 2006.

   [DTNSOV]   S. Farrell, S. Symington, H. Weiss, "Delay-Tolerant
                 Networking Security Overview", Work in Progress, October
                 2006.

   [DBFJHP04]   M. Demmer, E. Brewer, K. Fall, S. Jain, M. Ho, R. Patra,
                "Implementing Delay Tolerant Networking", Intel Research
                Technical Report IRB-TR-04-020, Dec. 2004.

   [RFC792]   Postel, J., "Internet Control Message Protocol", STD 5,
                RFC 792, September 1981.

   [RFC894]   Hornig, C., "A Standard for the Transmission of IP
                Datagrams over Ethernet Networks", STD 41, RFC 894, April
                1 1984.

   [RFC2960]   Stewart, R., Xie, Q., Morneault, K., Sharp, C.,
                Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M.,
                Zhang, L., and V. Paxson, "Stream Control Transmission
                Protocol", RFC 2960, October 2000.

   [RFC4088]   Black, D., McCloghrie, K., and J. Schoenwaelder, "Uniform
                Resource Identifier (URI) Scheme for the Simple Network
                Management Protocol (SNMP)", RFC 4088, June 2005.

   [S05]   K. Scott, "Disruption Tolerant Networking Proxies for
                On-the-Move Tactical Networks", Proc. MILCOM 2005
                (unclassified track), Oct. 2005.

   [T02]   W. Thies, et al., "Searching the World Wide Web in Low-
                Connectivity Communities", Proc. WWW Conference (Global
                Community track), May 2002.

12. Acknowledgments

Authors' Addresses

   Dr. Vinton G. Cerf
   Google Corporation
   Suite 384
   13800 Coppermine Rd.
   Herndon, VA 20171
   Phone: +1 (703) 234-1823
   Fax: +1 (703) 848-0727
   EMail: vint@google.com

   Scott C. Burleigh
   Jet Propulsion Laboratory
   4800 Oak Grove Drive
   M/S: 179-206
   Pasadena, CA 91109-8099
   Phone: +1 (818) 393-3353
   Fax: +1 (818) 354-1075
   EMail: Scott.Burleigh@jpl.nasa.gov

   Robert C. Durst
   The MITRE Corporation
   7515 Colshire Blvd., M/S H440
   McLean, VA 22102
   Phone: +1 (703) 983-7535
   Fax: +1 (703) 983-7142
   EMail: durst@mitre.org

   Dr. Kevin Fall
   Intel Research, Berkeley
   2150 Shattuck Ave., #1300
   Berkeley, CA 94704
   Phone: +1 (510) 495-3014
   Fax: +1 (510) 495-3049
   EMail: kfall@intel.com

   Adrian J. Hooke
   Jet Propulsion Laboratory
   4800 Oak Grove Drive
   M/S: 303-400
   Pasadena, CA 91109-8099
   Phone: +1 (818) 354-3063
   Fax: +1 (818) 393-3575
   EMail: Adrian.Hooke@jpl.nasa.gov

Dr. Keith L. Scott
The MITRE Corporation
7515 Colshire Blvd., M/S H440
McLean, VA 22102
Phone: +1 (703) 983-6547
Fax: +1 (703) 983-7142
EMail: kscott@mitre.org

Leigh Torgerson
Jet Propulsion Laboratory
4800 Oak Grove Drive
M/S: 238-412
Pasadena, CA 91109-8099
Phone: +1 (818) 393-0695
Fax: +1 (818) 354-6825
EMail: ltorgerson@jpl.nasa.gov

Howard S. Weiss
SPARTA, Inc.
7075 Samuel Morse Drive
Columbia, MD 21046
Phone: +1 (410) 872-1515 x201
Fax: +1 (410) 872-8079
EMail: howard.weiss@sparta.com

Please refer comments to dtn-interest@mailman.dtnrg.org. The Delay
Tolerant Networking Research Group (DTNRG) web site is located at
http://www.dtnrg.org.