Delay/Disruption Tolerant Networking
Internet-Draft
Intended status: Informational
Expires: 8 January 2026

D. V. G. Cerf
Google Corporation
S. C. Burleigh
JHU Applied Physics Laboratory
R. C. Durst
The MITRE Corporation
D. K. Fall
Intel Research, Berkeley
D. K. L. Scott
The MITRE Corporation
L. Torgerson
Jet Propulsion Laboratory
H. S. Weiss
SPARTA, Inc.
July 2025

Delay-Tolerant Networking Architecture
draft-cerf-dtn-4838bis-00

Abstract

This document describes an architecture for delay-tolerant and disruption-tolerant networks. It 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 overlay protocol that interconnects multiple internets. 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 IETF DTN working group and has been widely reviewed by that group.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://edbirrane.github.io/4838bis/draft-cerf-dtn-4838bis.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-cerf-dtn-4838bis/.

Discussion of this document takes place on the Delay/Disruption Tolerant Networking mailing list (mailto:dtn@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/dtn/. Subscribe at https://www.ietf.org/mailman/listinfo/dtn/.

Source for this draft and an issue tracker can be found at https://github.com/edbirrane/4838bis.

Status of This Memo

working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 2 January 2026.

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 encompass multiple divergent sets 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 lengthy signal propagation latencies\; 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].

Our motivation for developing 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], and "Revisiting the Use of the IP Protocol Stack in Deep Space" [SB01] further discusses the motivation for the development of DTN.

Briefly, 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 end-to-end retransmission from the data source based on timely and stable feedback from the data receiver is an effective means for repairing errors

* that the rate of end-to-end data loss is relatively small

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

* 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 no end-to-end paths may ever exist)\; do not require retransmission from the data source.

* Provide security mechanisms that protect the infrastructure from unauthorized use by discarding traffic as quickly as possible and secure data at rest as well as data in transit.

* Provide coarse-grained classes of service, flexible 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.

As such, the intent of the architecture is to enable deployment of a Solar System-wide network characterized by the scope of automation that was developed for the Internet, enabling comparable power and scalability.

To this end we define a new end-to-end protocol stack layer which we term "bundle layer". We propose a protocol, named Bundle Protocol (BP) [RFC9171], that functions at the bundle layer. The bundle layer forms a delay- and disruption-tolerant overlay that interconnects networks that can ordinarily sustain continuous end-to-end connectivity and brief round-trip times, employing persistent storage to help combat network interruption. 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. DTN also includes a number of diagnostic and management features and has a security model, optionally enabled, aimed at protecting infrastructure from unauthorized use.

The bundle layer provides functionality similar to the internet layer of interconnected gateways described in the original ARPANET/Internet designs [CK74]. It differs from ARPANET gateways, however, in that it is layer-agnostic and is focused on message forwarding rather than packet switching. However, both architectures 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. The architecture’s delay tolerant forwarding is comparable to the procedures of 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.

Separately, but also noteworthy, the DTN architecture can help to "flatten" the protocol stack. The Bundle Protocol’s scope of operations can be amplified by the insertion of additional "extension blocks" (such as the Block Integrity Block and Block Confidentiality Block defined for BP security {BPSEC}) in the headers of BP protocol data units and by the definition of administrative records (e.g., bundle status reports) that enable BP itself to accomplish network management functions. These features enable DTN network stack functionality to be augmented easily without the insertion of additional protocol layers.

Note that the use of the bundle layer is guided not only by its own

design principles but also by 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 5.16.

2. Conventions used in this document

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].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

3. Terminology

The following terms are used in the context of DTN in this document:

Bundle: A bundle is a protocol data unit of the Bundle Protocol *BP), so named because negotiation of the parameters of a data exchange may be impractical in a delay-tolerant network: it is often better practice to "bundle" with a unit of application data all metadata that might be needed in order to make the data immediately usable when delivered to the application. Each bundle comprises a sequence of two or more "blocks" of protocol data, which serve various purposes.

Block: A Bundle Protocol block is one of the protocol data structures that together constitute a well-formed bundle.

Application Data Unit: An application data unit (ADU) is the unit of data whose conveyance to the bundle’s destination is the purpose for the issuance of some bundle .

Bundle payload: A bundle payload (or simply "payload") is the content of the bundle’s payload block.

Bundle node: A bundle node is any entity that can send and/or receive bundles.

Bundle Protocol Agent: The Bundle Protocol Agent (BPA) of a node is that component of a bundle node that offers BP services to applications and executes the procedures of the Bundle Protocol.

Convergence-layer adapter: A convergence-layer adapter (CLA) is a node component that sends and receives bundles on behalf of the BPA, utilizing the services of some "integrated" protocol stack that is supported in one of the networks within which the node is functionally located. The topmost layer of such a stack is termed the "convergence layer".

Administrative record: A BP administrative record is an ADU that is exchanged between nodes for some BP administrative purpose. The only administrative record defined in this document is the status report, discussed later.

Bundle endpoint: A bundle endpoint (or simply "endpoint") is a set of zero or more bundle nodes that all identify themselves for BP

purposes by some common identifier, called a "bundle endpoint ID" (or, in this document, simply "endpoint ID").

Registration: A registration is the state machine characterizing a given node’s membership in a given endpoint.

Delivery: A bundle is considered to have been delivered at a node subject to a registration as soon as the ADU that is the payload of the bundle, together with any relevant metadata (an implementation matter), has been presented to an application in a manner consistent with the state of that registration.

Forwarding: To forward a bundle to a node is to invoke the services of one or more CLAs in a sustained effort to cause a copy of the bundle to be received by that node.

4. Providing the DTN Overlay Service

Earlier sections summarized the motivation for the DTN architecture and identified the design principles that guide its definition. This section describes the communication service that users can expect from the DTN-based network.

A DTN-enabled application sends messages of arbitrary length, here termed Application Data Units or ADUs [CT90], the delivery of which is necessarily constrained by the nature of the network. ADUs may not always be received in the order in which they were transmitted. Multiple parts of an ADU might arrive at the destination at different times.

ADUs are encapsulated by the bundle layer within protocol data units called "bundles", which are forwarded by DTN "nodes."

A DTN node (or simply "node" in this document) is an engine for sending and receiving bundles -- an implementation of the bundle protocol. 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"\; that is, a DTN endpoint is 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 contain one node (unicast), any one of the nodes in a group of nodes (anycast), or all nodes in a group of nodes (multicast). A single node may be contained in the MRGs of multiple endpoints.

While a bundle protocol endpoint is simply a set of DTN nodes, a given endpoint may be defined to contain zero node (the "null endpoint"), exactly 1 node (a "singleton endpoint", or N (where 0 <= N) nodes (enabling multicast endpoints).

Each bundle contains an originating timestamp, a useful life indicator ("time to live"), and a payload 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.

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 need to store bundles for minutes, hours, or days. We anticipate that most DTN nodes will use some form of persistent storage for this purpose -- disk, flash memory, etc. -- and that stored bundles will survive system restarts.

4.1. Quality of Service

Postal Service provides a reasonable metaphor for the Quality of Service (QoS) parameters that might be applied in 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 reliability, priority, and security.

However, in DTN the values of bundles' QoS parameters are subject to change as the bundles traverse the network. They are determined at the bundle forwarding nodes based on policy (as established by node administration) and on the immutable features of bundles such as source and/or destination.

While QoS parameters are thus not addressed in the design of the bundle protocol itself, several principles of QoS determination that may apply to DTN have been identified.

4.1.1. Custody Transfer

"Custody Transfer" is an architectural concept first articulated in the course of designing the CCSDS File Delivery Protocol. The concept is that the environments addressed by DTN are so different from terrestrial internetworking as to motivate a relaxation of the End-to-End Argument that governs the Internet protocols. Retransmission of lost bundle data from the source DTN node may take so much time (due to lengthy signal propagation latencies and punctuated connectivity) that delivery prior to time-to-live expiration will simply fail\; retransmission instead from the node that most recently forwarded the lost bundle can sharply reduce delivery latency. This principle -- the advance from forwarding node to forwarding node, along the end-to-end path, of lost data recovery functionality -- is termed "custody transfer."

Custody transfer in DTN is typically accomplished not by means of a dedicated protocol but rather by the use of reliable protocols in the underlying network(s) on which the bundle protocol overlay is hosted. The selection and invocation of these underlying protocol capabilities is a key element in DTN QoS provisioning.

4.1.2. Priority

The DTN architecture is well suited to supporting _relative_ measures of priority (low, medium, high), similar to Postal Service classes that address customers' sense of ADU delivery urgency: * 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.

The combination of useful life limit (as aserted by the application)

and policy as applied at DTN nodes can be used to determine how messages are forwarded and which routing algorithms are in use, thereby affecting the overall likelihood and timeliness of ADU delivery.

In this context, the applicable priority class of a bundle might have effect only in relation to the forwarding of other bundles from the same source and/or the same destination. This means that a high priority bundle from one source might not be delivered sooner (or with some other superior quality of service) than a medium priority bundle from a different source. It is likely, though, that a high priority bundle from one source to one destination will be handled in preference to a lower priority bundle sent from the same source and to the same destination.

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.

4.1.3. Postal-Style Delivery Options

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.

We have identified five delivery options. An application sending an ADU (the "subject ADU") may request any combination of the following options for the bundle issued in response to the application's request to send the subject ADU:

* Custody Transfer Requested - requests that the bundle be forwarded with reliability enhanced by custody transfer as described above. The bundle should be transmitted from one node to the next using reliable underlying transport protocols (if available).

* 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 option is similar to the postal "return-receipt" service.

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

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

* Report When Bundle Deleted - requests a Bundle Deletion Status Report be generated if and when the bundle is deleted at a DTN node. This is likewise designed primarily for diagnostic purposes.

The first of these delivery options is intended for routine 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 [RFC9172] are also enabled, then three additional delivery options become available:

* Confidentiality Required - requires that some subset of the blocks of the bundle be made secret from parties other than the sources

and destinations of those blocks.

* Authentication Required - requires that some subset of the blocks of the bundle be cryptographically authenticated. See Section Section 5.12 for more details.

* Error Detection Required - requires that any modification of some subset of the blocks of the bundle be detectable by cryptographic means.

5. DTN Architecture

This section presents a description of the major features of the architecture resulting from design decisions as guided by the design principles discussed above.

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

Each bundle has a defined format comprising 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 contain information typically found in the header and payload portions of protocol data units in other protocols.

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. A bundle may 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 essential element of the bundle-based style of forwarding is that bundles must 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 an additional network resource that must be managed and protected. Much of the research in DTN revolves around exploring these issues. Congestion is discussed in Section 5.11\; security mechanisms, including methods for DTN nodes to protect themselves from handling unauthorized traffic from other nodes, are discussed in [DTNSOV] and [RFC9172].

5.2. 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 the specification for the URI scheme of an EID, a node is able to determine the MRG of the DTN endpoint named by the EID. Accordlingly, an EID may identify zero, 1, or multiple DTN nodes,

enabling multicast destination or "report-to" endpoints. Each node is required to be a member of at least one endppoint in order to receive bundles\; each node is required to be a member of at least one "singleton" endpoint (that is, an endpoint that always has exactly one member) in order to source bundles that can be authenticated by the network.

When referring to a group of size greater than one, the delivery semantics may be of either the anycast or multicast variety. For anycast group delivery, a bundle is intended to be delivered to one node among a group of potentially many nodes\; for multicast delivery it is intended to be delivered to all nodes in the group, subject to the normal DTN maximum useful lifetime semantics.

The assertion of a node's membership in an endpoint is called a "registration" and in general is maintained persistently by a DTN node. This allows node registration information to survive operating system restarts.

Applications send ADUs destined for endpoints identified by EIDs, and they 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.

The assertion of an application's desire to receive ADUs destined for a particular EID is termed an "enrollment." Enrollments, too, may be maintained persistently by a DTN node, an implementation matter\; this would enable application enrollment information likewise to survive application and operating system restarts.

An application's attempt to establish an enrollment is not guaranteed to succeed. For example, an application could request enrollment in the reception of ADUs carried by bundles destined for an Endpoint identified by an EID that is uninterpretable or unavailable to the DTN node servicing the request. Such requests are likely to fail.

5.3. 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, separated from it by a colon (":") character, 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 [RFC3986]. Section 3.1 of [RFC3986] provides guidance on the syntax of scheme names.

The use of URIs to identify endpoints introduces a great deal of flexibility in the structuring of EIDs. DTN EIDs might, for example, be constructed based on DNS names, or might look like "expressions of interest" or take on the forms of database-like queries as in a directed diffusion-routed network [IGE00] or in intentional naming [WSBL99]. Being the names of sets of nodes, 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 might 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 DTN EID 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 [ANY-MULTI-CAST] for more details.

URIs are constructed based on rules specified in [RFC3986], using the US-ASCII character set. However, note this excerpt from Section 1.2.1 of [RFC3986], 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".

5.4. 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 invalid or even impossible. Furthermore, the 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.

5.5. Primary Bundle Fields

The bundles carried between and among DTN nodes obey a standard bundle protocol specified in [RFC9171]. Here we provide an overview of most of the fields carried in every bundle. The protocol is designed with a mandatory primary block, a mandatory 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:

* 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 at which an application requested that the ADU be sent (not when the corresponding bundle is sent into the network). DTN nodes are assumed to have a basic time synchronization capability (see Section 5.10).

* Lifespan - the length of time, following bundle creation tim, after which the bundle's ADU is no longer useful. The "expiration time" of a bundle is the time computed as the sum of the bundle's

creation time and its lifespan. If a bundle is stored at a network node (including the source's DTN node) when its expiration time is reached, it may be discarded.

* Bundle Processing Flags - indicates the selected delivery options. See Section 4.1.3.

* Source EID - EID of a singleton endpoint of which the bundle's source (the first sender) is the sole member.

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

* Report-To Endpoint ID - an EID identifying where bundle status reports should be sent. This may or may not identify the same endpoint as the Source EID.

5.6. Other Bundle Fields

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 [RFC9171] for additional details.

5.7. 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 commonly 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 be characterized by 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 magnitude of 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].

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.

5.7.1. 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 be instantiated 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 time at which a contact is established is delay-dependent. For example, a single scheduled contact between Earth and Mars would not begin 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 indeterminant 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].

5.8. Reliability and Custody Transfer

The most basic service provided by the bundle layer is unacknowledged, prioritized (but not guaranteed) unicast message delivery. The bundle protocol also supports two options for enhancing delivery reliability: custody transfer (described earlier) and application acknowledgment requests. Applications wishing to implement their own end-to-end message reliability mechanisms may respond to an application acknowledgment request in an application-specific manner.

5.9. 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 lengthy and highly variable round-trip times, 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 by providing support for routing and reliability, enabling existing 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 composed 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.

5.10. Timestamps and Time Synchronization

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

Bundle identification and expiration are supported by placing a creation timestamp and an explicit expiration field (expressed in milliseconds after the source timestamp) in each bundle. The origination timestamps on arriving bundles are made available to consuming applications in ADUs they receive by the application programming interface. Each bundle is tagged a timestamp unique to the source node’s EID. The EID and timestamp together uniquely identify a bundle. Unique bundle identification is used for a number of purposes, notably including bundle status reporting.

5.11. Congestion and Flow Control at the Bundle Layer

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 limited freedom in handling the situation. It can discard a bundle immediately upon reception if reception resources are insufficient to accommodate that bundle, possibly sending a bundle status report noting that the bundle was discarded. If storage resources are available elsewhere in the network, it may be able to make use of them in some way for bundle storage.

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 controlling 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] or QUIC [RFC9000]) might be preferable in such circumstances.

Congestion control in DTN is an ongoing research topic.

5.12. Security

The possibility of severe resource scarcity in some delay-tolerant networks dictates that some form of authentication and control over access 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 [RFC9172] that supports both integrity checking (which, when key-based, provides authentication as well) and confidentiality. These measures may be exercised at forwarding points along the bundle’s path ("hop-by-hop") and/or at the final destination node. While end-to-end use of BPSEC provides authentication for a principal such as a user (of which there may be many), hop-by-hop integrity checking is intended to intercept bogus data before its forwarding wastes the resources of the network.

In accordance with the goals listed above, DTN nodes discard traffic as early as possible if authentication 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 [RFC9172].

5.13. Administrative Records: Bundle Status Reports

As noted earlier, the DTN architecture accommodates the transmission and reception of administrative records, whose sources and destinations are the bundle protocol agents themselves. Administrative records correspond (approximately) to messages in the ICMP protocol in IP [RFC792]. The source EID of an administrative record is the administrative EID of the source node\; for EIDs formed in the "ipn" URI scheme, this is the EID whose service number is zero.

At this time, one class of useful administrative records has been defined: "bundle status reports" (BSRs). Bundle status reports are used to report status information or error conditions related to the bundle layer. They are directed to the report-to EIDs of the bundles to which they pertain, which might differ from the source EIDs. In some cases, arrival of a single bundle or bundle fragment may elicit multiple bundle status reports (e.g., in the case where a bundle is replicated for multicast forwarding).

The following BSRs are currently defined (also see [RFC9171] for more details):

* Bundle Reception - sent when a bundle tagged with the Report When Bundle Received option arrives at a DTN node. Generation of this message may be limited by local policy.

* Bundle Forwarded - sent when a bundle tagged with the 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 tagged with the Report When Bundle Deleted is discarded. This can happen for any of several reasons, including expiration. Generation of this message may be limited by local policy.

* Bundle Delivery - sent from a final recipient’s (destination) node when a bundle tagged with the Report When Bundle Delivered option is delivered. Generation of this message may be limited by local policy.

Every bundle status report must reference a received bundle. For this purpose, bundles are uniquely identified as discussed above.

5.14. 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.

5.14.1. Application Registration State

In long/variable delay environments, an asynchronous application interface seems most appropriate. Such interfaces may 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 enrollment state.

Application enrollment 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 enrollment 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 an enrollment and thereby establish application enrollment state, an application specifies an Endpoint ID at which it wishes to receive ADUs. This operation is somewhat analogous to the bind() operation in the common sockets API.

5.14.2. 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, which notionally includes a contact graph as described earlier), while forwarding makes use of state that is 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 priority class, 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 remains an active research topic.

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.

5.14.3. Security-Related State

The DTN security approach described in [RFC9172], 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. The cached information items may have expiration times, in which case the timely delivery of new security state information before current information expires is a critical responsibility of network management.

In addition, 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.

5.15. 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, what types of convergence layer (see Section 5.17) and routing protocols are in use, how bundles of differing priority classes should be scheduled, where and for how long bundles and other data are stored, what status reports may be generated and at what rate, etc.

5.16. 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 over 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.

5.17. Convergence Layer Considerations for Use of Underlying Protocols

Implementation experience with the DTN architecture has revealed an important architectural consideration 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 "convergence-layer adapters" which offer transmission functionality to the overlying bundle protocol and utilize the transmission functionality offered by the underlying protocol. The convergence layer adapters 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 adapter may vary substantially from another, depending on the type of underlying protocol it utilizes. For example, a TCP/IP convergence layer adapter for use in the Internet might only have to add message boundaries to TCP streams, whereas a convergence layer adapter 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 layer adapters implement protocols above and beyond the basic bundle protocol specified in [RFC9171], 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]).

6. 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 recognizes postal mail as a model of service classes and delivery semantics. It introduces a relaxation of the End-to-End Principle, offering assured end-to-end delivery by relying on the retransmission capabilities of underlying protocols operating between forwarding points on the end-to-end path. It accommodates many different forms of connectivity, including scheduled, predicted, and opportunistically connected delivery paths. 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.

7. Security Considerations

Security is an integral concern for the design of the Delay Tolerant Network Architecture, but its use is optional. Sections Section 4.1.3 and Section 5.14.3 of this document present some factors to consider for securing the DTN architecture, but separate documents [DTNSOV] and [RFC9172] define the security architecture in more detail.

8. 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].

9. References

9.1. Normative References

[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/rfc/rfc2119>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/rfc/rfc3986>.

9.2. Informative References

[AF03] Alonso, J. and K. Fall, "A Linear Programming Formulation of Flows over Time with Piecewise Constant Capacity and Transit Times", June 2003.

[ANY-MULTI-CAST] Zhao, W., Ammar, M., and E. Zegura, "Multicast in Delay Tolerant Networks", 2005.

[CK74] Cerf, V. and R. Kahn, "A Protocol for Packet Network Intercommunication", May 1974.

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

[DBFJHP04] Demmer, M., Brewer, E., Fall, K., Jain, S., Ho, M., and R. Patra, "Implementing Delay Tolerant Networking", December 2004.

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

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

[FHM03] Fall, K., Hong, W., and S. Madden, "Custody Transfer for Reliable Delivery in Delay Tolerant Networks", July 2003.

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

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

[IPN01] Internet Society Interplanetary Chapter, "InterPlaNetary Chapter", n.d., <http://www.ipnsig.org>.

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

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

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

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

[LFC05] Leguay, J., Friedman, T., and V. Conan, "DTN Routing in a Mobility Pattern Space", 2005.

[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, DOI 10.17487/RFC2960, October 2000, <https://www.rfc-editor.org/rfc/rfc2960>.

[RFC4088] Black, D., McCloghrie, K., and J. Schoenwaelder, "Uniform Resource Identifier (URI) Scheme for the Simple Network Management Protocol (SNMP)", RFC 4088, DOI 10.17487/RFC4088, June 2005, <https://www.rfc-editor.org/rfc/rfc4088>.

[RFC792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/rfc/rfc792>.

[RFC894] Hornig, C., "A Standard for the Transmission of IP Datagrams over Ethernet Networks", STD 41, RFC 894, DOI 10.17487/RFC0894, April 1984, <https://www.rfc-editor.org/rfc/rfc894>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/rfc/rfc9000>.

[RFC9171] Burleigh, S., Fall, K., and E. Birrane, III, "Bundle Protocol Version 7", RFC 9171, DOI 10.17487/RFC9171, January 2022, <https://www.rfc-editor.org/rfc/rfc9171>.

[RFC9172] Birrane, III, E. and K. McKeever, "Bundle Protocol Security (BPSec)", RFC 9172, DOI 10.17487/RFC9172, January 2022, <https://www.rfc-editor.org/rfc/rfc9172>.

[S05] Scott, K., "Disruption Tolerant Networking Proxies for On-the-Move Tactical Networks", October 2005.

[SB01] Burleigh, S., "Revisiting the Use of the IP Protocol Stack in Deep Space", September 2024.

[SB03] Burleigh, S., Hooke, A., Torgerson, L., Fall, K., Cerf, V., and B. Durst, "Delay-Tolerant Networking - An Approach to Interplanetary Internet", IEEE Communications Magazine , July 2003.

[T02] Thies, et al, W., "Searching the World Wide Web in Low-Connectivity Communities", May 2002.

[WJMF05] Wang, Y., Jain, S., Martonosi, M., and K. Fall, "Erasure Coding Based Routing in Opportunistic Networks", 2005.

[WSBL99] Adjie-Winoto, W., Schwartz, E., Balakrishnan, H., and J. Lilley, "The Design and Implementation of an Intentional Naming System", December 1999.

Acknowledgments

Authors’ Addresses

Dr. Vinton G. Cerf
Google Corporation
13800 Coppermine Rd. Suite 384
Herndon, VA
USA
Email: vgcerf@gmail.com

Scott C. Burleigh
Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Rd, Laurel, MD 20723
USA
Email: sburleig.sb@gmail.com

Robert C. Durst
The MITRE Corporation
7515 Colshire Blvd., M/S H440
McLean, VA
USA

Email: durst@mitre.org

Dr. Kevin Fall
Intel Research, Berkeley
2150 Shattuck Ave.
Berkeley, CA
USA
Email: kfallca@gmail.com

Dr. Keith L. Scott
The MITRE Corporation
7515 Colshire Blvd., M/S H440
McLean, VA
USA
Email: kscott@mitre.org

Leigh Torgerson
Jet Propulsion Laboratory
4800 Oak Grove Drive, M/S 238-412
Pasadena, CA
USA
Email: jordan.l.torgerson@jpl.nasa.gov

Howard S. Weiss
SPARTA, Inc.
7075 Samuel Morse Drive
Columbia, MD
USA
Email: howard.weiss@parsons.com