A Delay-Tolerant Network Architecture for Challenged Internets

Kevin Fall
Intel Research, Berkeley
kfall@intel-research.net

ABSTRACT

The highly successful architecture and protocols of today's Internet may operate poorly in environments characterized by very long delay paths and frequent network partitions. These problems are exacerbated by end nodes with limited power or memory resources. Often deployed in mobile and extreme environments lacking continuous connectivity, many such networks have their own specialized protocols, and do not utilize IP. To achieve interoperability between them, we propose a network architecture and application interface structured around optionally-reliable asynchronous message forwarding, with limited expectations of end-to-end connectivity and node resources. The architecture operates as an overlay above the transport layers of the networks it interconnects, and provides key services such as in-network data storage and retransmission, interoperable naming, authenticated forwarding and a coarse-grained class of service.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Computer Networks, Network Architecture

General Terms: Algorithms, Design, Security

1. INTRODUCTION

The existing TCP/IP based Internet service model provides end-to-end inter-process communication using a concatenation of potentially dissimilar link-layer technologies. The standardization of the IP protocol and its mapping into network-specific link-layer data frames at each router supports interoperability using a packet-switched model of service. Although often not explicitly stated, a number of key assumptions are made regarding the overall performance characteristics of the underlying links in order to achieve this service: an end-to-end path exists between a data source and its peer(s), the maximum round-trip time between any node pairs in the network is not excessive, and the end-to-end packet drop probability is small. Unfortunately, a class of *challenged networks*, which may violate one or more of the assumptions, are becoming important and may not be well served by the current end-to-end TCP/IP model. Examples include:

• Terrestrial Mobile Networks: Some of these networks may become unexpectedly partitioned due to node mobility or changes in signal strength (e.g. RF interference), while others may be partitioned in a periodic, predictable manner. For example, a commuter bus could act as a store and forward message switch with only limited-range RF communication capability. As it travels from place to place, it provides a form of message switching service to its nearby clients to communicate with distant parties it will visit in the future.

SIGCOMM'03, August 25-29, 2003, Karlsruhe, Germany.

• Exotic Media Networks: Exotic communication media includes near-Earth satellite communications, very long-distance radio or optical links (e.g. deep space communications with light propagation delays in the seconds or minutes), acoustic links in air or water, and some free-space optical communications. These systems may be subject to high latencies with predictable interruption (e.g. due to planetary dynamics or the passing of a scheduled ship), may suffer outage due to environmental conditions (e.g. weather), or may provide a predictably-available store-and-forward network service that is only occasionally available (e.g. low-earth orbiting satellites that "pass" by periodically each day).

• Military Ad-Hoc Networks: These systems may operate in hostile environments where mobility, environmental factors, or intentional jamming may be cause for disconnection. In addition, data traffic on these networks may have to compete for bandwidth with other services at higher priority. As an example, data traffic may have to unexpectedly wait several seconds or more while high-priority voice traffic is carried on the same underlying links. Such systems also may also have especially strong infrastructure protection requirements.

• Sensor/Actuator Networks: These networks are frequently characterized by extremely limited end-node power, memory, and CPU capability. In addition, they are envisioned to exist at tremendous scale, with possibly thousands or millions of nodes per network. Communication within these networks is often *scheduled* to conserve power, and sets of nodes are frequently named (or addressed) only in aggregate. They typically employ "proxy" nodes to translate Internet protocols to the sensor network native protocols.

Given the large accumulated experience and number of systems compatible with the TCP/IP protocols, it is natural to apply the highly successful Internet architectural concepts to these new or unusual types of networks. While such an application is conceivable, the effects of very significant link delay, non-existence of end-to-end routing paths, and lack of continuous power or large memory at end nodes present substantial operational and performance challenges to such an approach. In some cases, unusually large bandwidth-delay products can also present difficulties, especially when high bandwidth efficiency is required.

In an effort to adapt Internet to unusual environments, one class of approaches attempts to engineer problem links to appear more similar to the types of links for which TCP/IP was designed. In effect, these approaches, which we term *link-repair approaches*, "fool" the Internet protocols into believing they are operating over a comparatively well-performing physical infrastructure. They strive to maintain the end-to-end reliability and fate sharing model of Internet, and generally require the use of IP in all participating systems.

Another common approach to deal with challenged networks is to attach them to only the *edge* of the Internet by means of a special proxy agent. This provides access to and from challenged networks from the Internet, but does not provide a general way to use such networks for data *transit*. Without supporting transit, the full capabilities of these networks may go unrealized. Indeed, supporting transit is often of particular interest because remotely-deployed conventional networks

(e.g. Intranets) may only be accessible through challenged intermediate networks for economic reasons.

In this paper, and its extended version [8], we argue that to achieve interoperability between very diverse networks, especially those engineered for extreme environments or that often suffer from network partitioning, link-repair approaches alone will not suffice and network-specific proxies are undesirable. Instead, we suggest a general purpose message-oriented reliable overlay architecture as the appropriate approach to tie together such networks, forming an "internetwork of challenged internets." The approach, which provides the service semantics of asynchronous message delivery, may be used in combination with TCP/IP where appropriate. Its design is influenced by the interoperability properties of the classical Internet design, the robust non-interactive delivery semantics of electronic mail, and a subset of the classes of service provided by the US Postal System. These networks have all evolved to become highly successful communication networks supporting millions of daily users.

2. CHALLENGED INTERNETS

Qualitatively, challenged internetworks are characterized by latency, bandwidth limitations, error probability, node longevity, or path stability that are substantially worse than is typical of today's Internet. We use the Internet's performance as a baseline due to its enormous scale and influence. This section explores the path properties, network architectures and end node resources found across the broad range of challenge networks introduced above and how they influence the design of a network architecture designed to accommodate them.

2.1 Path and Link Characteristics

High Latency, Low Data Rate: If we temporarily disregard processing and queuing delays (we return to queuing delays shortly), the transmission and propagation delays of a link are directly affected by the underlying transmission medium. For some challenged networks, transmission rates may be comparatively small (e.g. about 10kbps for underwater acoustic modems and low-power radios in sensor nodes) and latencies may be comparatively large (to about a second or two). Also, data rates may be largely asymmetric (e.g. remote instruments may have a comparatively high rate downlink channel for relaying telemetry but a small uplink used for device control). In some extreme cases, no return channel at all may be available (e.g. communication with some military assets requiring covert operation such as submarines).

Disconnection. In many challenged networks, end-to-end disconnection may be more common than connection. Generally speaking, disconnection may be broadly categorized as due to a fault or not. Faults have been studied extensively for conventional networks, and will not be further considered here. Non-faulty disconnections arise most frequently in wireless environments, from primarily two sources: motion and low-duty-cycle system operation. Disconnection due to motion may be highly predictable (e.g. satellite passes, busses that act as data routers, etc) or opportunistic (nodes arrive in communication range due to random walk) and may arise due to motion of either end-nodes, routers, or some other object or signal that obscures the communication. Disconnection due to low-duty-cycle operation is common among low-capability devices (e.g. sensor networks), and is often predictable. Exceptional conditions requiring immediate attention (event responses) can perturb the otherwise periodic low-duty-cycle operation at unpredictable times.

Long Queuing Times. For multi-hop paths in conventional packet networks with statistical multiplexing, the queuing time often dominates the transmission and propagation delays. Queuing time rarely exceeds a second (and is typically much less) and packets are discarded at routers if next-hop neighbors are not instantaneously reachable. In contrast for networks where disconnection may be common, the queuing time could be extremely large by comparison (hours, perhaps days). Furthermore, source-initiated retransmission may be extremely expensive due to the limited number of transmission opportunities. Combined, these issues suggest that messages may need to be stored for potentially long periods of time at (message) routers.

2.2 Network Architectures

Interoperability Considerations In most challenged networks, the network "architectures" consist primarily of a link and media-access control protocol, and are not designed with interoperability (or very large scale) in mind. The reason for this is that in many cases, merely communicating *at all* over some links is still an active area for research, and the desire to use such links in an internetwork has not yet become a primary focus. Thus, these networks tend to be comparatively simple and local in scope, and may fail to provide even the baseline abstractions that are well-matched for supporting layered protocol families (such as Internet). Implementations frequently "cut corners" when targeted for deployment on memory and power-limited devices, mixing together data from various system functional blocks into messages that are difficult to dis-aggregate. They also frequently fail to implement reliability, congestion control, and security.

Security. In challenged networks where communication media is frequently oversubscribed, link capacity is a precious resource and access to the "service" of data forwarding should be protected by some authentication and access control mechanism, at least at critical points in the topology. If multiple classes of service (CoS) are available, some mechanism to control access to them is also likely to be required. In such cases, an approach to security which only involves the endpoints is not very attractive, stemming from two issues. First, end-to-end-only approaches typically require some exchange of challenges or keys, which would be undesirable for high-delay and disconnection-prone networks. Secondly, it is undesirable to carry unwanted traffic all the way to its destination before an authentication and access control check is performed. The later problem has been (and remains) a problem for the Internet, but in that case the issue is significantly worse because of the desire for small end-to-end delays.

2.3 End System Characteristics

Limited Longevity In some challenged networks, end nodes are placed in hostile environments. This is especially true for sensor networks, military networks, and networks of devices used by emergency response personnel. In such cases, network nodes may not last long, due to environmental dangers or power exhaustion. If such networks also remain disconnected for long periods of time, it is entirely possible that the round-trip or even one-way delivery time of a particular message may exceed the sending node's lifetime. In such cases it is useless to utilize conventional end-to-end acknowledgment schemes to help verify delivery. Rather, the responsibility for reliable delivery should be *delegated* to some other party, and any notification of successful or unsuccessful delivery needs to be delivered to a delegate that remains operational.

Low Duty Cycle Operation. When nodes are deployed in areas lacking power infrastructure, their communication patterns are often scheduled *a-priori*. In some cases (e.g. battery powered sensors), duty cycles of well under 1% are desirable in order to achieve a reasonable longevity of the entire network. Such devices would typically collect data at some periodic rate, and report it at some (perhaps less frequent) rate. For these types of networks, transmission scheduling, in concert with path selection, raises special considerations for routing.

Limited Resources In several of the challenged network examples above, nodes with limited memory and processing capability are used. Consider, for example, an instrument with limited memory tasked with acquiring sensor readings of some random physical phenomena. It is undesirable to prohibit the instrument from collecting further samples because its memory

is fully utilized with copies of in-transit data. In addition, the amount of time the end-nodes will need to keep retransmission buffers is at least the round-trip-time times the expected number of retransmissions (plus 1), which can be large for high latency and/or lossy paths. While the node may be able to implement a powered down mode of operation during this interval, provided it has nonvolatile storage, doing so considerably complicates the system design, particularly if other asynchronous messages may have to be received or unexpected physical events of interest occur. This example suggests that if reliability is to be incorporated into the network design, end nodes should be provided a way to empty their retransmission buffers comparatively quickly, and to not necessarily have to wait for an end-to-end acknowledgment.

3. FIX THE INTERNET PROTOCOLS?

3.1 PEPs, Boosters and Proxies

To combat the various problems with the Internet protocols over challenged networks (or to enhance their performance over subnetworks with special features), several types of in-network entities (so-called "middle boxes") have been developed that modify protocol behavior. Investigations of link repair approaches, primarily for satellite and terrestrial wireless Internet access via TCP/IP, have resulted in the development of Performance Enhancing Proxies (PEPs) [2] and *protocol boosters* [9]. These agents, which actively modify the end-to-end data stream, in effect "fool" TCP/IP-based end stations into operating more efficiently over paths involving links with poor or unusual performance.

Use of PEPs is discouraged (by the IETF [2]) except for particular environments where they are necessary for ''reasonable'' performance. This restriction is due to their fragility. In particular, they may contain state which is necessary for connection operation (thereby violating the Internet *fate sharing* principles [7] which suggest connection state should reside only in end stations), they confound end-to-end diagnostics and reliability by (partially) changing the communicating endpoint, they significantly increase system complexity if mobility is frequent (due to the need to migrate state when end-nodes move), and many require both directions of data to flow through the PEP (a problem if asymmetric routing is used). They also pose a significant challenge for end-to-end security mechanisms implemented below the transport layer such as IPSEC. While protocol boosters are conceived with the idea that they are entirely transparent to end-protocols, this assumption limits their overall ability to improve performance when subnet conditions are especially bad (e.g. disconnected).

An alternative to boosters and PEPs involves application-layer proxies that provide a specialized Internet-to-"special network" name mapping and protocol translation. Proxies are generally used at the edge of such special networks, and allow interoperability with the Internet without requiring IP routers to exist inside. This approach is important, as there is often significant reluctance to deploy IP protocols inside these challenged networks due to concerns of overhead, address management, or protocol implementation difficulties.

The disadvantage of proxies are in their specificity. Proxies usually use one of two approaches: they respond to a specialized set of commands specific to the special network, or act as raw data conduits. The first approach limits the ability to re-use the proxies for different applications; the second method fails to take advantage of any special resources the proxy node may have to offer (such as storage or processing capabilities), and requires applications communicating with the proxy to employ specialized protocols that are compatible with those of the special networks'. Finally, no general inter-proxy routing capability is currently used, meaning that if any dynamic routing is performed between proxies it is specific to the types of proxies in question. It would be generally more attractive to standardize on a set of proxy-based services which provide I/O to and *through* the challenged network, if possible, using a common set of methods.

3.2 Electronic Mail

Electronic mail, an asynchronous message delivery system, provides an abstraction that comes close to addressing many of the problems posed by challenged networks. In particular, flexible naming, asynchronous message-based operation, and in-band error reporting are particularly useful. In addition, it has been shown to operate over a rich set of network technologies (an especially important feature prior to the widespread use of the Internet standard electronic mail).

Email falls short due to its lack of dynamic routing, weakly-defined delivery semantics, and lack of consistent application interface. With respect to routing, existing approaches rely on a (statically preconfigured) set of mail relays which provide very little tolerance to network outages. The delivery semantics of electronic mail appear to be *mostly reliable* delivery with *likely* failure notification. Messages can fail to be delivered due to mis-addressing, persistent lack of intermediate or end-node storage, failure of underlying transport protocols, or enforcement of policies on content (e.g. content filtering or size restrictions). When delivery succeeds, end-to-end acknowledgments are generally not provided automatically. Upon failure, the original message and accumulated errors are generally returned to the sender, possibly with additional information supplied to a third party. While this diagnostic information is extremely useful, the typical end-user has little direct ability to correct the problem.

3.3 Motivation for an Additional Architecture

While proxies, PEPs, and electronic mail can help to deal with some of the problems posed by challenged internetworks, they do not provide a complete solution. Disconnected paths, limited-capability/longevity end devices with potentially specialized protocol stacks, and unusual routing (including predictable or periodic connections) appears to preclude (or at least pose serious difficulties for) the direct use of IP's addressing and routing features. Furthermore, its forwarding function (which drops packets if a next-hop route is not immediately available) is problematic for frequently-disconnected links. With respect to reliable data delivery, Internet's idea of fate sharing suggests that per-connection state should remain only in end-stations, because a failure of one of them would presumably render the data connection essentially useless. In many challenged environments, this assumption does not hold. For example, it may be quite useful to allow a node to "hand off" its end-node connection state if it has other tasks to accomplish, particularly if it is power or memory limited. Doing so would not violate fate sharing entirely (per connection state would not be required in all intermediate nodes), but would represent a somewhat different fate sharing behavior than is implemented in the current Internet.

In addition to the problems associated with the network itself, applications designed with assumptions of low delay can also encounter problems when operated over challenged networks. While it would be unfair to fault application designers for not contemplating high delay and disconnection, some guidance as to what APIs are appropriate for these situations is appropriate. Indeed, it may be advantageous to provide applications with a direct indication as to whether to expect ordinary or extraordinary delays and allow them to customize their behaviors accordingly [19].

Given the assumptions, the most desirable framework for supporting challenged internets would appear to be a network service and API providing a sort of least common denominator interface: non-interactive messaging. Based on experience with the Internet, we conclude such a system should combine some overlay routing capability such as is present in peer-to-peer systems with the delay-tolerant and disconnection-tolerant properties of electronic mail. If implemented at the application layer (in the form of a proxy), such a system could conceivably provide a gateway function between radically dissimilar networks. These considerations together motivate the articulation of a new architecture, which we now describe.

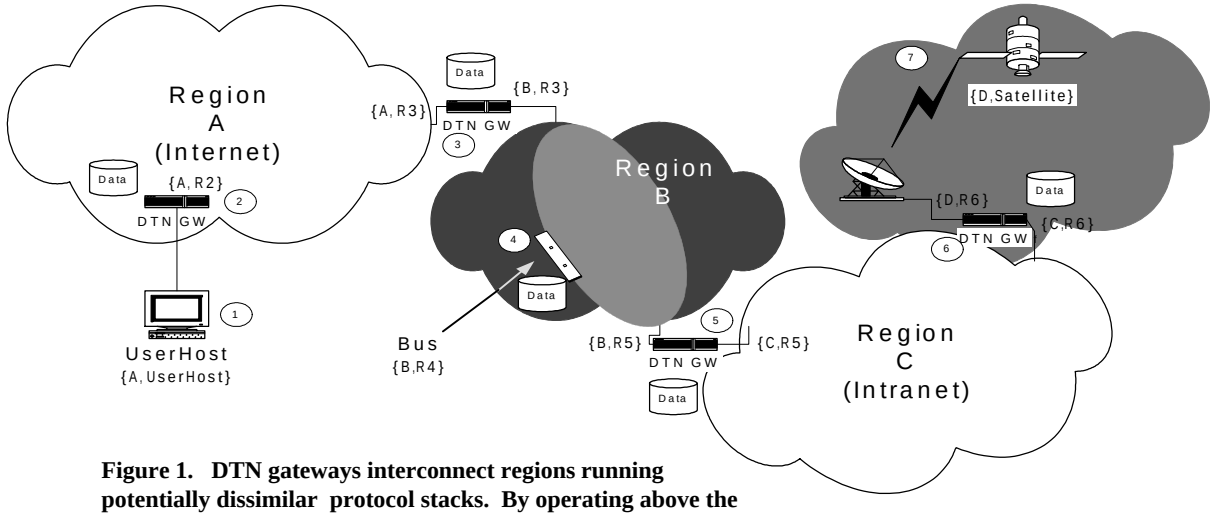

Figure 1. DTN gateways interconnect regions running potentially dissimilar protocol stacks. By operating above the transport protocols in use on the incident networks, they provide virtual message switching, in-network retransmission, and name mapping, allowing globally-interoperable names to be mapped

4. A DELAY TOLERANT MESSAGE BASED OVERLAY ARCHITECTURE

The architecture proposed here for interoperability between and among challenged networks is called the *delay tolerant networking* architecture (DTN), and is based on an abstraction of message switching. Message aggregates are known as "bundles" and are adopted from [3]. The routers that handle them are called "bundle forwarders" or DTN gateways.

As an "overlay" architecture, DTN is intended to operate above the existing protocol stacks in various network architectures and provide a store-and-forward gateway function between them when a node physically touches two or more dissimilar networks. For example, within the Internet the overlay may operate over TCP/IP, for deep space links it may provide a gateway service to CFDP [5], and in delay-tolerant sensor/actuator networks it may provide interconnection with some yet-to-be-standardized sensor transport protocol. Each of these networking environments have their own specialized protocol stacks and naming semantics developed for their particular application domain. Achieving interoperability between them is accomplished by special DTN gateways located at their interconnection points.

4.1 Regions and DTN Gateways

The DTN architecture includes the concepts of *regions* and *DTN gateways*, as illustrated in Figure 1. In this example, four regions are illustrated (A, B, C, D). Region B includes a DTN gateway resident on a commuter bus[1] that cycles between DTN gateways 3 and 5. Region D includes a low-earth-orbiting satellite link (LEO) that also provides periodic connectivity (albeit perhaps more regular than the bus which may be subject to vehicular congestion or other delays).

Region boundaries are used as interconnection points between dissimilar network protocol and addressing families. More formally, two nodes are in the same region if they can communicate without using DTN gateways (generally using existing protocols local to the containing region). We expect a small number of *region types* (e.g. Internet-like, ad-hoc mobile, periodic disconnected, etc.) may evolve and each instance of the same type will implement a similar stack of underlying protocols. DTN gateways correspond to both the Metanet "waypoint" concept in [22] and also to the definition of gateways described in the original ARPANET design [4]. The waypoint concept describes a point through which data must pass in order to gain entry to a region. This point can serve as a basis for both translation (between region-specific encodings) as well as a point to enforce policy and control.

A DTN gateway spanning two regions consists logically of two "halves," each half in one of the adjacent regions above their corresponding transport protocols, analogous to ARPANET-style gateways structured above specific link layer protocols. In operating above the transport layer, however, DTN gateways differ from ARPANET gateways and are instead focused on reliable message routing instead of best-effort packet switching. DTN gateways are responsible for storing messages in nonvolatile storage when reliable delivery is required and mapping between differing transports by resolving globally-significant *name tuples* to locally-resolvable names for traffic destined internally to an adjacent region (see following section). They also may perform authentication and access control checks on arriving traffic to ensure forwarding is to be allowed.

4.2 Name Tuples

For routing of DTN messages, we elect to use identifiers for objects or groups of objects called *name tuples* comprising two variable length portions. In Figure 1, the DTN name tuple(s) for each end-point and each router "half" is illustrated in curly braces in the form {Region Name, Entity Name}. The first portion is a globally-unique, hierarchically structured region name. It is interpreted by DTN gateways to find the path(s) to one or more DTN gateways at the edge of the specified region. It is populated into DTN forwarding tables either statically (by a network administrator), or by one or more dynamic DTN-layer routing protocols (which could be computed centrally for a region, for example). A region name's hierarchical structure provides the ability to reduce the size of DTN forwarding tables in a fashion similar to the Internet's route aggregation in CIDR [6], yet allows for additional flexibility due to the variable-length substrings allowed between the hierarchy delimiters. Note that despite their similar appearance to DNS names, region names need not necessarily be resolved to any form of address or resolved in a distributed hierarchy as DNS names are.

[1] Utilizing a bus as a data router is not purely hypothetical. In some parts of the world it is a more economically viable and reliable method for transporting data if high delays can be tolerated. See http://www.daknet.net or http://www.dtnrg.org.

The second portion identifies a name resolvable within the specified region and need not be unique outside the region. As illustrated in the figure, it may be of arbitrary structure and may contain special symbols resolvable in only the origin or destination regions. In the case of the Internet, for example, we could have the following tuple:

{internet.icann.int, "http://www.ietf.org/oview.html"}

This tuple would refer to the Internet region (in some yet-to-be-defined region hierarchy), along with an Internet-specific local identifier (in this case, a Universal Resource Identifier or *URI*; see [13] for more details). As a message transits across a (potentially long and heterogeneous) collection of regions, only its region identifier is used for routing. Upon reaching the edge of the destination region, the entity name information is locally-interpreted, and translated if necessary, into a protocol-standard name (or address) appropriate to the containing region. This method of resolving names results in a form of *late binding* for tuples in which only the portion of the tuple immediately needed for message forwarding (the region portion) is used by DTN gateways. By not imposing any particular fixed structure on the second portion of a tuple, any reasonable naming scheme can be easily accommodated, even unusual ones (e.g. treating only sensor aggregates as endpoints as in [11]). The concept of late binding has been used in other systems. For example in [21], it is used primarily for supporting anycast where a location-independent service discovery operation is desired.

Late binding of tuples in DTN differs from the DNS-style Internet naming and addressing scheme which requires one or more DNS transactions to complete *prior* to the start of an Internet end-to-end conversation. For challenged networks, the need to consult a name-to-address mapping that may be resident only in the destination region seems impractical given potentially large end-to-end delays.

The choice of adopting names rather than addresses as the basis for labeling participants in the end-to-end routing scheme derives from an observation of recent trends in the operation of the Internet. The Internet design makes frequent reference to resource sharing as enabled by a (distributed) interprocess control mechanism. Addresses are used for routing and for referring to a computational resource (i.e. server endpoint), and naming is added to make the addressing easier for humans. Today's Internet includes objects such as search engines and page caches which are used extensively. In many cases, a name (in the form of a URI or URL web address) effectively refers to a query for data rather than identification of a particular end-system computational resource that provides it.

4.3 A Postal Class of Service

The notion of a challenged network inherently implies a limitation on various resources. Priority-based resource allocation is therefore important to adopt in the overall model, but care must be taken to avoid so burdensome a class of service architecture as to have it be unimplementable or confusing to users. The approach taken here is to adopt a subset of the types of services provided by the US Postal Service. This system has evolved to meet the needs of millions of users exchanging non-interactive traffic and has the added benefit of already being reasonably familiar to most users. As such, it seems a highly compelling starting point for considering the classes of service to be offered by a primarily non-interactive networking architecture.

Over its roughly 230 year history, the Post Office Department (and the modern US Postal System of the last half-century) has developed a remarkable class of service offering associated with the seemingly straightforward service of mail delivery. In addition to the basic delivery categories of first-class, priority, express mail, parcel post and "bound printed matter," a large variety of special delivery options are available. As anyone who has utilized these special delivery operations can attest, some combinations of options are not supported, whereas others have mutual interdependence. The complexity of this system seems too great as a basis for a network class of service offering, as several of the options are not directly applicable to a data network (e.g. air delivery) or are tied to financial considerations that are considered to be out of scope for the DTN design (e.g. insurance). Yet the postal classes of service are compelling due to their familiarity and long history. In a distilled form, therefore, the following core postal services seem to be attractive due to their coarse granularity and intuitive character: low, ordinary, and high priority delivery; notifications of mailing, delivery to the receiver (return receipt), and route taken (delivery record). The model is extended with the option of reliable delivery (somewhat akin to careful handling), and messages requiring this service are handled somewhat differently by the routing system in that they require persistent storage and a *custody transfer* at each routing hop (see below, Part 4.5).

4.4 Path Selection and Scheduling

The DTN architecture is targeted at networks where an end-to-end routing path cannot be assumed to exist. Rather, routes are comprised of a cascade of time-dependent *contacts* (communication opportunities) used to move messages from their origins toward their destinations. Contacts are parameterized by their start and end times (relative to the source), capacity, latency, endpoints, and direction. In addition, a measure of a contact's predictability can help to choose next-hop forwarders for message routing as well as select the next message to be sent. The predictability of a route exists on a continuum ranging from completely predictable (e.g. wired connection or a periodic connection whose phase and frequency are well-known) to completely unpredicted (an "opportunistic" contact in which a mobile message router has come into communication range with another DTN node). Note that the measure of a contact's predictability is sensitive to its direction. For example, a dial-up connection may be completely predictable from the initiator's point of view while being completely unpredicted from the callee's point of view.

The particular details of path selection and message scheduling are expected to be heavily influenced by region-specific routing protocols and algorithms. At this relatively early stage of DTN development, several challenging problems have been identified: determination of the existence and predictability of contacts, obtaining knowledge of the state of pending messages given assumptions of high delay, and the problem of efficiently assigning messages to contacts and determining their transmission order. While very simple (e.g. greedy) heuristics for these problems can be implemented without excessive problems, each issue represents a significant challenge and remains as future work. A linear programming formulation of the (idealized) routing/scheduling problem with contacts has been described recently in [1].

4.5 Custody Transfer and Reliability

The DTN architecture includes two distinct types of message routing nodes: persistent (P) and non-persistent (NP). P nodes are assumed to contain nontrivial amounts of persistent message store, and NP nodes might not. Unless they are unable or unwilling to store a particular message, P nodes generally participate in *custody transfer* using the appropriate transport protocol(s) of the containing region. A custody transfer refers to the acknowledged delivery of a message from one DTN hop to the next and the corresponding passing of reliable delivery responsibility. Custody transfer is akin to delegating responsibility for delivering postal mail to a person or service that promises (or contracts) to do so.

The custody transfer concept is fundamental to the architecture in order to combat potentially high loss rates and to relieve potentially resource-poor end nodes from responsibilities related to maintaining end-to-end connection state. In particular, end-nodes do not ordinarily need to keep a copy of data that has been custodially transferred to a DTN next hop. For end nodes *insisting* on an end-to-end acknowledgment, a

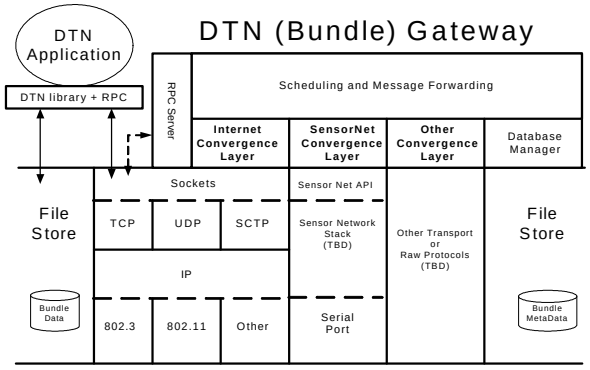

Figure 2. Structure of a DTN gateway. Multiple *convergence layers*, one per protocol stack, provide a common interface to the message scheduler/forwarder.

"delivery confirmation" may be optionally requested, although how to respond to this indication is left to the requesting application.

In contemplating a change from end-to-end reliable delivery semantics to a hop-by-hop reliability approach (with end-to-end notification), we may ask whether a less robust type of reliability is being provided by the hop-by-hop approach. We believe that the custody transfer mechanism is not necessarily *less* reliable than using typical end-to-end reliability, although it is *different*. This opinion stems from the observation that in many circumstances, where end nodes cannot be assumed to remain operational for long periods of time, that the chances for data to be reliably delivered using delegation can exceed its changes of being successfully delivered end-to-end. Furthermore, the provision of the end-to-end optional acknowledgment is consistent with the *end-to-end argument* [16] ---that only the applications truly know what they require. Indeed, custody transfer can be viewed as a performance optimization for end-to-end reliability that involves the movement of the endpoint.

4.6 Convergence Layers and Retransmission

The facilities provided by the transport protocols in use within the regions containing a DTN P node may vary significantly. For example, any transport protocol may or may not offer the following: reliable delivery, connections (with indications of connection failure), flow control, congestion control, and message boundaries. As the bundle forwarding function assumes an underlying reliable delivery capability with message boundaries when performing custody transfer, transport protocols lacking these features must be appropriately augmented. Figure 2 illustrates the implementation structure for a bundle forwarder, including a number of transport-protocol-specific *convergence layers* used to add reliability, message boundaries, and other features above those transport protocols requiring augmentation. (Note that TCP in the Internet requires augmentation due to its lack of message boundaries; SCTP, which includes support for message boundaries, would likely use a minimal convergence layer and not require much augmentation). The design of most convergence layers is specific to the transport protocols being augmented, and are therefore beyond the scope of this paper.

In cases where reliable delivery is provided by an underlying transport, the corresponding convergence layer need only manage connection state and initiate restarts if a connection is lost. In the case of connection-oriented protocols, detection of a lost connection is generally provided through the application interface (via signals or other errors using the socket interface, for example). In cases where no direct support is provided for detecting failures, the bundle forwarding function itself may set a coarse-grained timer to re-start message transfers should it be concluded that they have failed. This is designed as a fallback measure in cases of underlying communication failure, and is not expected to be an especially efficient mechanism for initiating retransmission.

The appropriate choice of timeout for the coarse-grain retransmission timer will vary depending on the details of the containing region, and thus represents a certain form of layer violation in which the overlay "network" layer is able to be sensitive to underlying "physical" layer properties by requesting information from a convergence layer. In challenged networks, knowledge of some path properties at the forwarding layer appears to be very useful in selecting error control policy. In particular, some rough expectation of the round-trip time is extremely useful to trigger attempted repair actions.

4.7 Time Synchronization

The DTN architecture requires a coarse level of (relative) time synchronization, which is used for identifying message fragments (see [8]) and also for purging messages that have exceeded their source-specified lifetimes. In most circumstances, however, there are several additional benefits derived from imposing a more stringent constraint on time synchronization (e.g. on the order of one millisecond). The motivation stems from the observation that synchronized timing is needed by many distributed applications used in challenged environments and is required by the DTN's approach to scheduling and path selection (in cases where contact start and end times are known ahead of time). In addition, given reasonably accurate time synchronization, DTN congestion management techniques can conceivably predict at what times congestion may abate. Although more burdensome than time synchronization requirements on the Internet (which are essentially nonexistent), we believe the problem of fine time synchronization is sufficiently well-developed as to be a default policy for most networks. Protocols such as NTP [14] have provided 1ms accurate time synchronization (or better) within the Internet for years, and most existing networks for extreme environments already provide some (often out-of-band) means for obtaining accurate time[2].

4.8 Security

The security requirements for the DTN architecture differs somewhat from traditional network security models in that the set of principals includes the network routers (i.e. DTN gateways) themselves in addition to the communicating endpoints. In the DTN case, we are likely to be more interested in verifiable access to the carriage of traffic at a particular class of service and want to avoid carrying traffic potentially long distances that is later found to be prohibited.

To implement the security model, each message includes an immutable "postage stamp" (a type of capability) containing a verifiable identity of the sender (or role), an approval (and approving authority) of the requested class of service (CoS) associated with the message, and other conventional cryptographic material to verify accuracy of the message content. Routers check credentials at each DTN hop, and discard traffic as early as possible if authentication fails. This approach has the associated benefit of making denial-of-service attacks considerably harder to mount as compared with conventional Internet routers.

The current approach uses public key cryptography as a starting point for keying. Routers and users are issued public/private keypairs, and a principal sending a message must obtain a signed copy of its public key from a certificate authority known to DTN forwarders. (All routers are assumed to be pre-equipped with copies of one or more certificate

[2] There is an effort to extend NTP to space, although this work is still under development [15].

authority public keys and their own public/private key pairs). A principal then presents its signed public key along with a message to be carried signed with the corresponding private key. At the first DTN router, the signed public key is used to validate the sender and requested class of service against an access control list stored in the gateway. Accepted messages are then re-signed in the key of the gateway for transit. Using this approach, only first-hop gateways need cache per-user certificates, and then only for adjacent users. Non-edge "core" gateways can rely on the authentication of upstream gateways to verify the authenticity of messages. We believe this approach will help to improve the scalability of key management for these networks, as it will limit the number of cached public key certificates to a function of the number of adjacent gateways rather than the number of end-users. This should provide both the obvious advantage of memory savings, but also an improvement to system management as gateway keys are expected to be changed less frequently than end-user keys. As DTN gateways are likely to be deployed in remote areas, re-keying may be a comparatively burdensome system management tasks, so limiting the number and frequency of certificate updates should provide additional savings.

The approach described above is partially susceptible to compromised routers. If an otherwise-legitimate router is compromised, it would be able to utilize network resources at an arbitrary CoS setting and send traffic purportedly originating from any user who's identity is known to the router. However, if the message signature is carried end-to-end (an option for DTN security), a legitimate user could repudiate the origin of any traffic generated in this manner at a later time. Thus, we believe a reasonable trade-off is to admit the possibility that a compromised router could launch a denial-of-service attack in order to gain the scalability benefits of not checking end-user credentials at every hop.

4.9 Congestion and Flow Control

As a form of hop-by-hop architecture, flow control and congestion control for DTN are closely related. Flow control in this context refers to limiting the sending rate of a DTN node to its next (DTN) hop. Congestion control refers to the handling of contention for the persistent storage at a DTN gateway. The mechanisms available to deal with these issues may be classified as proactive or reactive. Proactive methods generally involve some form of admission control, to avoid the onset of congestion in the first place. In many cases, a region may be under the administrative control of a single entity, and this approach may be practical. If proactive methods are insufficient or unavailable, reactive means (most likely involving direct flow control) must be used which usually result in degraded performance when the actual operational delays are high.

Two aspects of the DTN architecture make the issue of congestion control especially challenging (as compared to other aspects of the architecture): contacts may not arrive for some time in the future (so accumulated data may not have an opportunity to drain for some time), and received messages for which custody has been accepted cannot be discarded except under extreme circumstances or on expiration. Given these constraints, the possibilities to handle congestion include reserving buffer space as a function of CoS, rejecting incoming connections for new messages when buffer space is full, arranging for custody transfers to other potential custodians that may not be the most desirable next hop (a form of hot potato routing) and discarding non-custody messages in favor of any requiring custody transfers. In unusual and dire circumstances, a facility for removing messages requiring custody may be available, but removing such information is to be avoided if at all possible, as deleting reliable bundles would be considered a system fault.

The current approach uses a shared priority queue for allocating custody storage. First, any expired messages are cleared. Arriving messages that are too large are denied custody transfer. Next, messages are spooled based on priority and useful lifetime (specified by the sender and carried in each message). Two potential problems that arise include a form of priority inversion (arriving higher-priority messages may not have custody storage available if lower-priority messages arriving earlier have been custodially received) and head-of-line blocking. The blocking can arise when a DTN gateway accepts custody for messages that are outgoing on a contact that has not yet started, and is subsequently asked to forward messages to a currently-available contact that does not require a custody transfer. In such a case, the persistent storage in the node may be completely consumed by the pending messages, thereby preventing the non-custody messages from transiting.

For implementation of flow control, a DTN forwarder will attempt to take advantage of whatever flow control mechanisms are present in the underlying region-local transport protocols. For most mature networks, some such mechanism exists already (e.g. TCP, X.25, RTS/CTS, XON/XOFF, explicit admission/rate control, etc). For other networks where such mechanisms are still being developed, region-specific mechanisms may be constructed in the DTN forwarders' convergence layers. Doing so is (naturally) region-specific, and is beyond the scope of this paper. In any case, the uppermost functions of a DTN forwarder generally assume the existence of flow control, so some such mechanism should be present to help ensure reliable message delivery.

5. APPLICATION INTERFACE

As described, the DTN architecture is built as an overlay network using messages as the primary unit of data interchange. Applications making use of the architecture must be careful not to expect timely responses and must generally be capable of operating in a regime where a request/response turn-around time exceeds the expected longevity of the client and server processes. In addition, applications must be prepared to handle the creation and manipulation of name tuples and their registrations (for demultiplexing received messages), class of service specifiers, and authentication information. The application interface is non-blocking, and callback registrations are persistent. Generally speaking, all DTN applications should be structured to continue operating in the face of reboots or network partitioning as much as possible.

6. RELATED WORK

The DTN architecture is based most closely on work that originated with the Interplanetary Internet [3] design, but represents a significant generalization to other types of networks suffering from non-Internet-like performance characteristics. It addresses several of the issues raised in the "network survivability" literature [18], especially with respect to networks lacking continuous connectivity.

With respect to store-and-forward routing in other frequently-disconnected networks, a number of recent efforts have arisen. In ZebraNet [12], wireless sensor nodes (attached to animals) collect location data and opportunistically report their histories when they come in radio range of base stations. They explore the case of mobile base stations and sensor devices and the use of a pair of flooding-based routing protocols. In DataMules [17], low-power sensor nodes can save power if periodically visited by a "mule" that travels among them and provides a non-interactive message store-and-forward service. In these two efforts plus that of Vahdat [20], mobility models are employed in simulation to predict the ability of partially connected networks to deliver data eventually.

The use of late binding for names in DTN is shared with, although not directly based upon, the work on Intentional Naming [21]. Here, names represent a form of query and are used specifically for anycast in order to locate nearby network services. Routing based on names is shared, to some degree, with Internet Content Routing [10]. This work focuses on using routing on names to provide a content distribution facility for the Internet, addressing its scalability and performance. It does not use two separate name components

as in DTN, but does suggest the viability of the name-based routing mechanism. The generality of the entity portion of names is influenced by [11], where database-like queries are effectively used as addresses for groups of sensor nodes.

The architectural thinking regarding interoperability and layering is guided by principles of the ARPANET/Internet [4,7]. DTN gateways operate in many ways similar to Internet routers, but are adapted for use in high-delay and disconnected environments by storing messages for potentially long periods of time.

7. CONCLUSION

The DTN architecture aims to provide interoperable communications between a wide range of networks which may have exceptionally poor and disparate performance characteristics. The design embraces the notion of message switching with in-network storage and retransmission, late-binding of names, and routing tolerant of network partitioning to construct a system better suited to operations in challenged environments than most other existing network architectures, particularly today's TCP/IP based Internet.

A prototype DTN implementation has been developed under the Linux operating system, which implements the application interface, basic forwarding across scheduled and "always on" connections, detection of new and lost contacts, and two convergence layers (for TCP/IP as well as a sensor network proxy). The prototype has been used as a proof-of-concept of the overall architecture, and also to show the general utility of the non-interactive reliable messaging service it provides.

The architecture represents a generalization of the Interplanetary Internet architecture to challenged networks other than space. The previous work was closely tied to issues of deep space communications in particular, but contributed many key ideas toward the development of a networking architecture applicable for challenged internetworks more generally. The design also derives in part from some interesting trends in the Internet: a move toward content-based naming, creation of administrative "regions", and alternative routing structures (e.g. network overlays).

The proposed DTN architecture advocates a change to the basic service model and system interface most Internet-style applications have become accustomed to, motivated by the exceptionally poor performance present in some networks. This is a comparatively radical approach; other approaches aim to "repair" underlying link performance problems or alter limited portions of the Internet architecture, such as routing, with additional protocols in an effort to keep the current service model and existing TCP/IP based protocols constant. Because it provides a different type of network service than Internet, the DTN design makes a different set of choices in the architectural design space: messages versus packets, a form of hop-by-hop reliability and security versus end-to-end, name based routing versus address based routing, and a routing abstraction of partially-connected rather than fully-connected network graph. Interestingly, DTN can be overlaid upon the TCP/IP based Internet easily, and therefore remains compatible. This is not the most interesting case, however, as its strength lies in its ability to tie together dramatically different types of networks with unusual connectivity properties. As such, in some ways it makes more limited assumptions on the underlying protocol layers than IP does upon its underlying link layers.

Only time will tell what application interfaces and service semantics will most appropriately match applications to challenged networks, but we believe the DTN architecture puts forth several design decisions worthy of consideration. In addition, we believe it is timely to consider a very broad range of network characteristics in formulating a new network architecture, as it appears likely an ever increasing number of these features will have to be dealt with.

8. ACKNOWLEDGMENTS

The author wishes to thank the members of the Interplanetary Internet Research Group for their previous work on the initial definitions of bundling and naming, without which this architecture would not exist. Members of this group include Vint Cerf (MCI), Adrian Hooke and Scott Burleigh (NASA/JPL), Bob Durst and Keith Scott (the MITRE Corporation), and Howard Weiss (SPARTA). The author is especially indebted to Bob Durst and Scott Burleigh for an ongoing collaboration regarding the DTN design. Versions of the manuscript benefited from the comments of David Culler, Sylvia Ratnasamy, the anonymous reviewers, David Hutchison and Sushant Jain.

9. REFERENCES

[1] 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

[2] J. Border et. al., "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations", Internet RFC3135, June 2001

[3] V. Cerf et. al., "Interplanetary Internet (IPN): Architectural Definition", http://www.ipnsig.org/reports/memo-ipnrg-arch-00.pdf

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

[5] CFDP Protocol Specification, CCSDS 727.0-B-1, Jan 2002, http://www.ccsds.org

[6] E. Chen, J. Stewart, "A Framework for Inter-Domain Route Aggregation", Internet RFC2519, Feb 1999

[7] D. Clark, "The Design Philosophy of the DARPA Internet Protocols", Proc. SIGCOMM 1988

[8] K. Fall, "A Delay-Tolerant Network Architecture for Challenged Internets", Intel Research Technical. Report IRB-TR-03-003, Feb 2003

[9] D. Feldmeier, A. McAuley, J. Smith, D. Bakin, W. Marcus, T. Raleigh, "Protocol Boosters", IEEE JSAC, Apr 1998

[10] M. Gritter, D. Cheriton, "An Architecture for Content Routing Support in the Internet", Proc. Usenix USITS, March 2001

[11] J. Heidemann et. al., "Building Efficient Wireless Sensor Networks with Low-Level Naming", Proc. SOSP, Oct 2001

[12] P. Juang, H. Oki, Y. Wang, M. Maronosi, L. Peh, D. Rubenstein, "Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet", Proc. ASPLOS, Oct 2002

[13] M. Mealling, R. Denenbers, eds., "Report from the Joint W3C/IETF URI Planning Interest Group: Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations", Internet RFC 3305, Aug 2002

[14] D. Mills, "Network Time Protocol (Version 3) Specification, Implementation and Analysis", Internet RFC1305, Mar 1992

[15] D. Mills, H. Nair, "Timekeeping in the Interplanetary Internet", *in progress*, http//:www.eecis.udel.edu/~mills/ipin.html

[16] J. Saltzer, D. Reed, D. Clark, "End-to-End Arguments in System Design", ACM Trans on Computer Systems, 2(4), Nov 1984

[17] R. Shah, S. Roy, S. Jain, W. Brunette, "Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks", IEEE SNPA Workshop, May 2003

[18] J. Sterbenz, et. al., "Survivable Mobile Wireless Networks: Issues, Challenges and Research Directions", WiSe 2002, Sep 2002

[19] J. Sterbenz, T. Saxena, R. Krishnan, "Latency-Aware Information Access with User-Directed Fetch Behaviour for Weakly-Connected Mobile Wireless Clients", BBN Tech. Report 8340, May 2002

[20] A. Vahdat, D. Becker, "Epidemic Routing for Partially-Connected Ad Hoc Networks", Duke Tech Report CS-2000-06, 2000

[21] W. Adgie-Winoto, E. Schwartz, H. Balakrishnan, J. Lilley, "The Design and Implementation of an Intentional Naming System", Proc. SOSP, Dec 1999

[22] J. Wroclawski, "The MetaNet: White Paper", Workshop on Research Directions for the Next Generation Internet", May 1997, http://www.cra.org/Policy/NGI/papers/wroklawWP