A Delay-Tolerant Network Architecture for Challenged Internets

Kevin Fall

Intel Research

Berkeley, CA

kfall@intel.com

http://www.intel-research.net

Aug 26, 2003 – SIGCOMM
Karlsruhe, Germany

Unstated Internet Assumptions

- Some path exists between endpoints
  - Routing finds (single) “best” existing route
    - [some exceptions…e.g. ECMP]
- End-to-end RTT is not terribly large
  - A few seconds at the very most (usually much less)
  - →window-based flow/congestion control works
- E2E reliability using ARQ works well (enough)
  - True for low loss rates (under 2% or so)
- Packets are the right abstraction
  - Internet (IP) makes packet switching interoperable
  - Routers don’t modify packets (much) when forwarding

New challenges…

- Very Large E2E Delays
  - Natural prop delay could be seconds to minutes
  - If disconnected, queuing times may be much longer
- Intermittent and *Scheduled* Links
  - Disconnection may not be due to failure (e.g. LEO sats and scheduling links down for power management)
  - Retransmission may be very expensive
    - Unauthorized access could be a big problem
- ‘Radically’ Heterogeneous Network Architectures
  - Many specialized networks won’t/can’t ever run IP

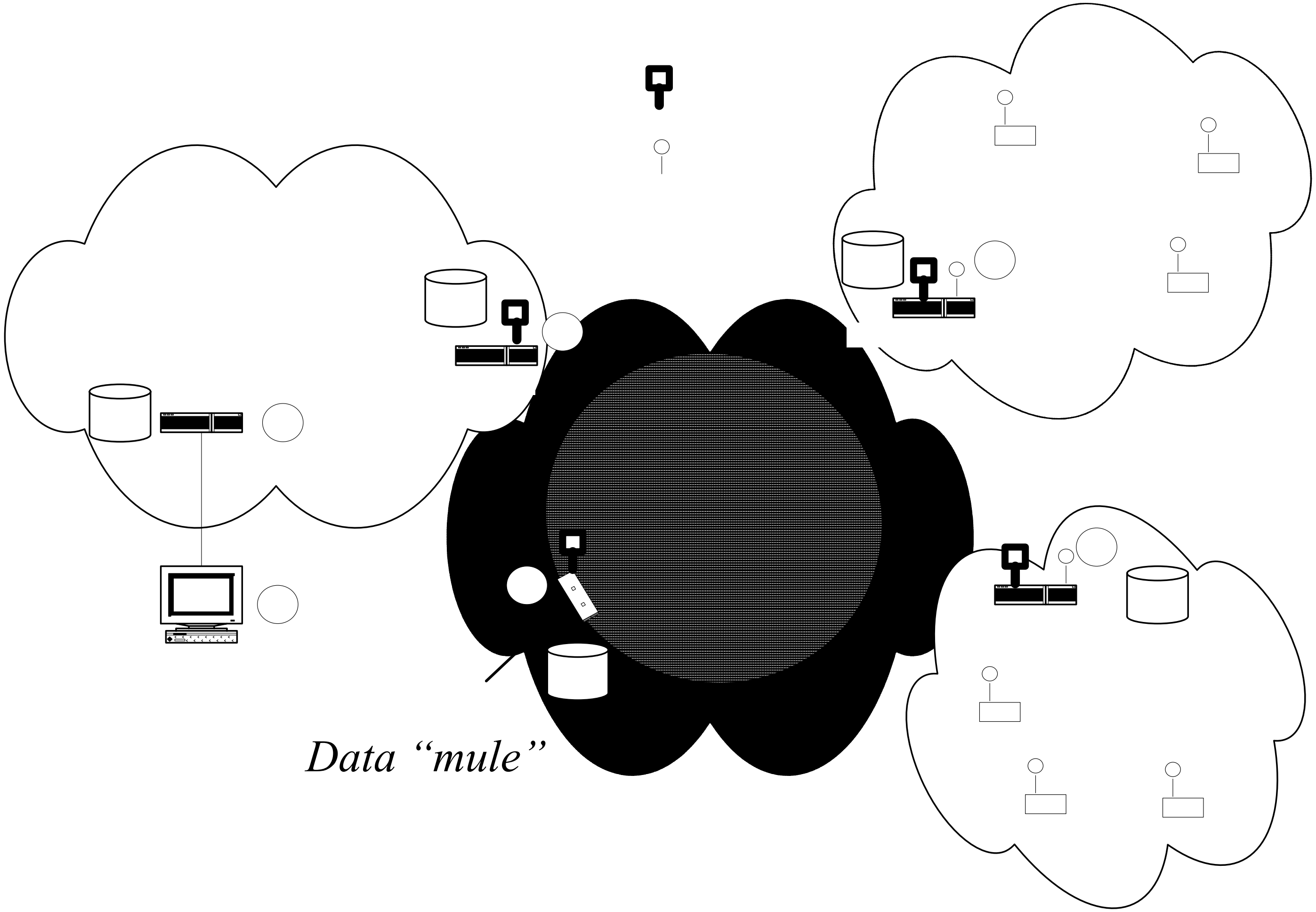
Delay-Tolerant Architecture

- Goals
  - Interoperability across network architectures
  - Reasonable performance in high loss/delay and frequently-disconnected environments
- Components
  - Flexible naming scheme with late binding
  - Message-based overlay abstraction (+API)
  - Routing and link/contact scheduling w/CoS
  - Per-(overlay)-hop authentication and reliability

Naming

- Names ("tuples") are of the URI form:
  - `bundles://`**<region-name>**`/`**<URI>**
  - Write this more simply as **(R,L)**
- Separates region (routing) from admin name
  - **R**: routing region [globally valid]
  - **L**: region-specific format, opaque outside region **R**
- Late binding of **L** permits naming flexibility
  - Routing based only on region portion
  - **L** could encode esoteric naming scheme [e.g. diffusion]
    - Could be object names, addresses, queries, etc.
  - Borrows from late binding in URLs and URIs

Example with Sensor Networks



Data

Home Base

Reliable Message Overlay

- End-to-End Reliable Message Service: ***bundles***
  - "postal-like" message delivery over regional transports
  - *Optional*: enhanced reliability, class of service, return receipt, and "traceroute"-like functions with 3rd-party "report-to" indicator
- Enhanced Reliability via ***Custody Transfer***
  - *Current Custodian* owns reliable-delivery promise
  - Bundles transferred between custodians toward destination in database-style transaction
  - Sender may free resources upon successful custody transfer (destination considered an eligible custodian)

Routing in a DTN

- Scheduled (known) / Unscheduled (opportunistic)
  - S/U characterization may be direction-specific
    - Consider the two ends of a user/ISP link
- Formulation as an LP (ideal case):
  - Minimize the *evacuation* time
  - Constraints on time, buffers, messages, priority
  - Several non-ideal options under investigation
- Predictability continuum:
  - Intermediate “predicted” category may evolve as a result of statistical estimation
  - Concept of *entropy* of a route [?]

Flow and Congestion Control

- FC is hop-by-hop in the overlay
  - Takes care of CC implicitly
  - Coarse timescale (e.g. ‘filesystem full’)
- FC for custody transfer not so easy:
  - Don’t want custody-traffic awaiting a contact to block forwarding of traffic to an available contact
  - *Options*: stop taking custody, separately queue custody and non-custody traffic, use destination queues, timeout
- Regional transport protocols may support FC
  - How to use built-in FC to effect bundle-layer FC?

Implementation and API

- DTN agent separated from client library
  - Both are RPC-based client and server
  - Either can be interrupted and restarted
- Client <–> agent association via register/callback
  - Registrations [and delivery actions] can be persistent
  - Can poll from last point on re-association
- Agent implements the ‘heavy lifting’:
  - DB for app (de)registrations, bundle send/recv/demux
  - Name resolution in destination region as required
  - Basic routing, scheduling and storage management functions
  - Custody transfer
  - Authentication and access control (if requested)

Status

- DTN is a message-oriented overlay for:
  - Internetworking in frequently-disconnected networks
  - Interconnecting ‘radically heterogeneous’ networks
- It evolved from the IPN Architecture
- There is a prototype implementation
  - ~20K lines of C code and some JAVA
  - Demonstrated as basis for query processing in disconnected sensor network
- There is an IRTF research group (DTNRG)

People

- People (designers and implementers):
  - Bob Durst, Keith Scott (MITRE)
  - Scott Burleigh (NASA/JPL)
  - (me)
- More people (vision, design, commentary):
  - Vint Cerf (MCI)
  - Adrian Hooke (NASA/JPL)
  - Juan Alonso (SICS)
  - Howard Weiss (SPARTA)
- The *dtn-interest* list and workshop participants

For more Information

- Delay Tolerant Networking Research Group
  - http://www.dtnrg.org
- Internet Research Task Force
  - http://www.irtf.org
- DTN Mailing list
  - dtn-interest@mailman.dtnrg.org
- Interplanetary Internet SIG (ISOC group)
  - http://www.ipnsig.org

www.dtnrg.org

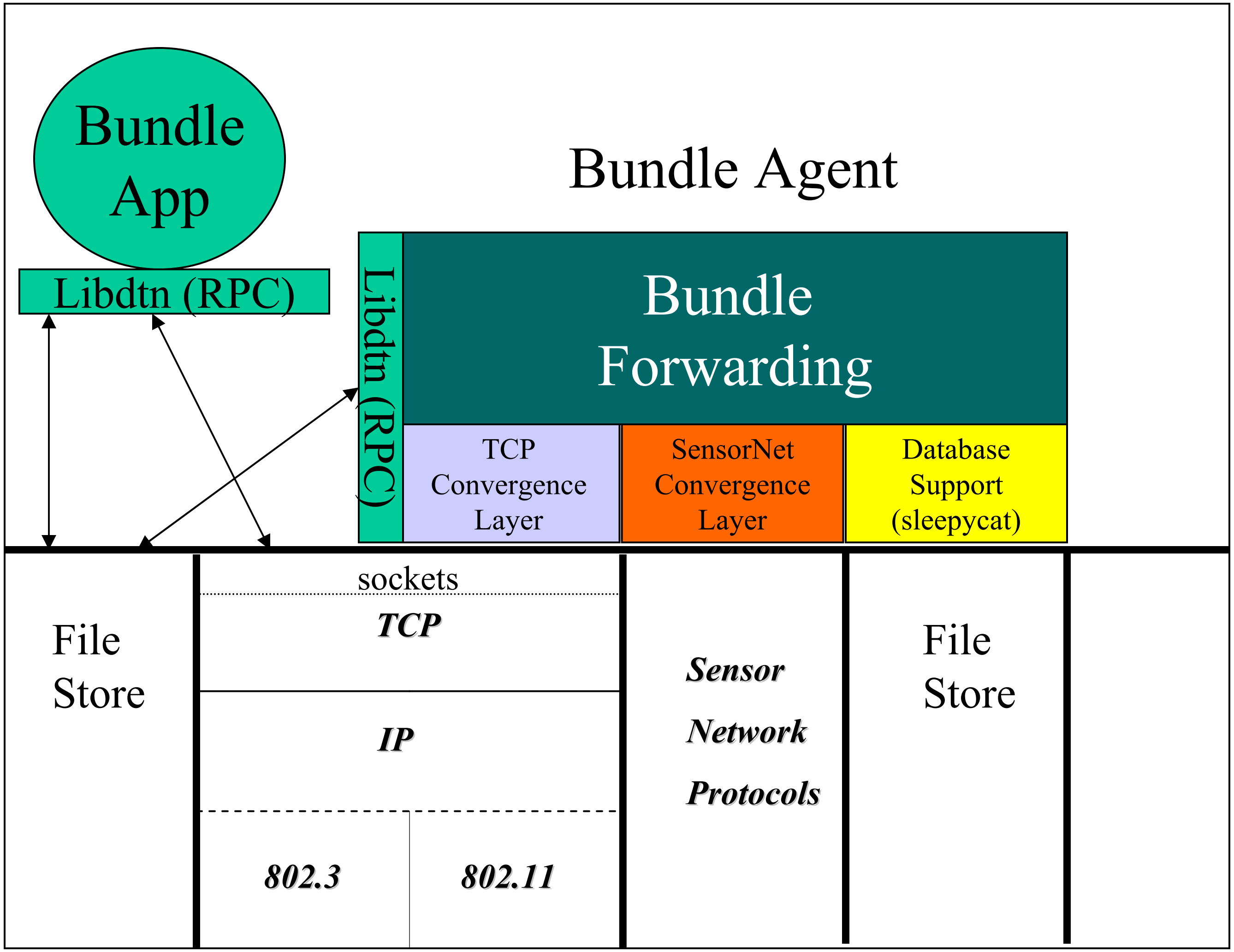
Thank you…

So, is this all just e-mail?

	naming/	routing	flow	multi-	security	reliable	priority
	late binding		contrl	app		delivery	
e-mail	Y	N	sort-of	sort-of	opt	Y	N(Y)
DTN	Y	Y	Y	Y	opt	opt	Y

- Many similarities to e-mail service interface
- Primary difference involves routing
- E-mail depends on an underlying layer’s routing:
  - Cannot generally move messages closer to their destinations in a partitioned network
  - In the Internet (SMTP) case, not delay tolerant or efficient for long RTTs due to “chattiness”
- E-mail security authenticates only user-to-user

Bundle Agent


Bundle App
Libdtn (RPC)
Bundle Agent
Libdtn (RPC)
Bundle Forwarding
TCP Convergence Layer
SensorNet Convergence Layer
Database Support (sleepycat)
File Store
sockets
TCP
IP
802.3
802.11
Sensor Network Protocols
File Store