Delay-Tolerant Networking for Challenged Internets

Kevin Fall

Intel Research

Berkeley, CA

kfall@intel-research.net

http://www.intel-research.net http://www.dtnrg.org

Apr 25, 2003 – UC Berkeley

intel

Unstated Internet Assumptions

- End-to-end RTT is not terribly large
  - A few seconds at the very most [typ < 500ms]
  - (reactive window-based flow/congestion control works)
- Some path exists between endpoints
  - Routing finds single “best” existing route
    - [ECMP is an exception]
- E2E Reliability using ARQ works well
  - True for low loss rates (under 2% or so)
- Packet switching is the right abstraction
  - Internet/IP makes packet switching interoperable

Non-Internet-Like Networks

- Stochastic mobility
  - Mesh networks
  - Mobile routers w/disconnection (e.g. ZebraNet)
- Periodic/predictable mobility
  - Spacecraft communications
  - Busses, mail trucks, police cars, etc (InfoStations)
- "Exotic" links
  - Deep space [40+ min RTT; episodic connectivity]
  - Underwater [acoustics; low rate; high error; latency]

New challenges…

- Very Large Delays
  - Natural prop delay could be seconds to minutes
  - If disconnected, may be much longer
- Intermittent/Scheduled/Opportunistic Links
  - Scheduled transfers can save power and help congestion; scheduling required for rare link assets
- High Link Error Rates / Low Capacity
  - RF noise, light or acoustic interference, LPI/LPD concerns
- Different Network Architectures
  - Many specialized networks won’t/can’t ever run IP

What to Do?

- Some problems surmountable in Internet
  - ‘cover up’ the link problems using PEPs
  - Mostly used at “edges,” not for transit
- Performance Enhancing Proxies (PEPs):
  - Do “something” in the data stream causing endpoint TCP/IP systems to not notice there are problemn
  - Lots of issues with transparency– security, operation with asymmetric routing, etc
- Some environments *never* have an e2e path…
  - And won’t ever run IP…

Delay-Tolerant Networking Architecture

- Goals
  - Internetwork(s) supporting interoperability across ‘radically heterogeneous’ networks
  - Acceptable performance in high loss/delay/error environments
  - Decent performance for low loss/delay/errors
- Components
  - Flexible Naming Scheme with late binding
  - Message Overlay Abstraction and API
  - Routing and link/contact scheduling w/CoS
  - Per-hop Authentication and Reliability

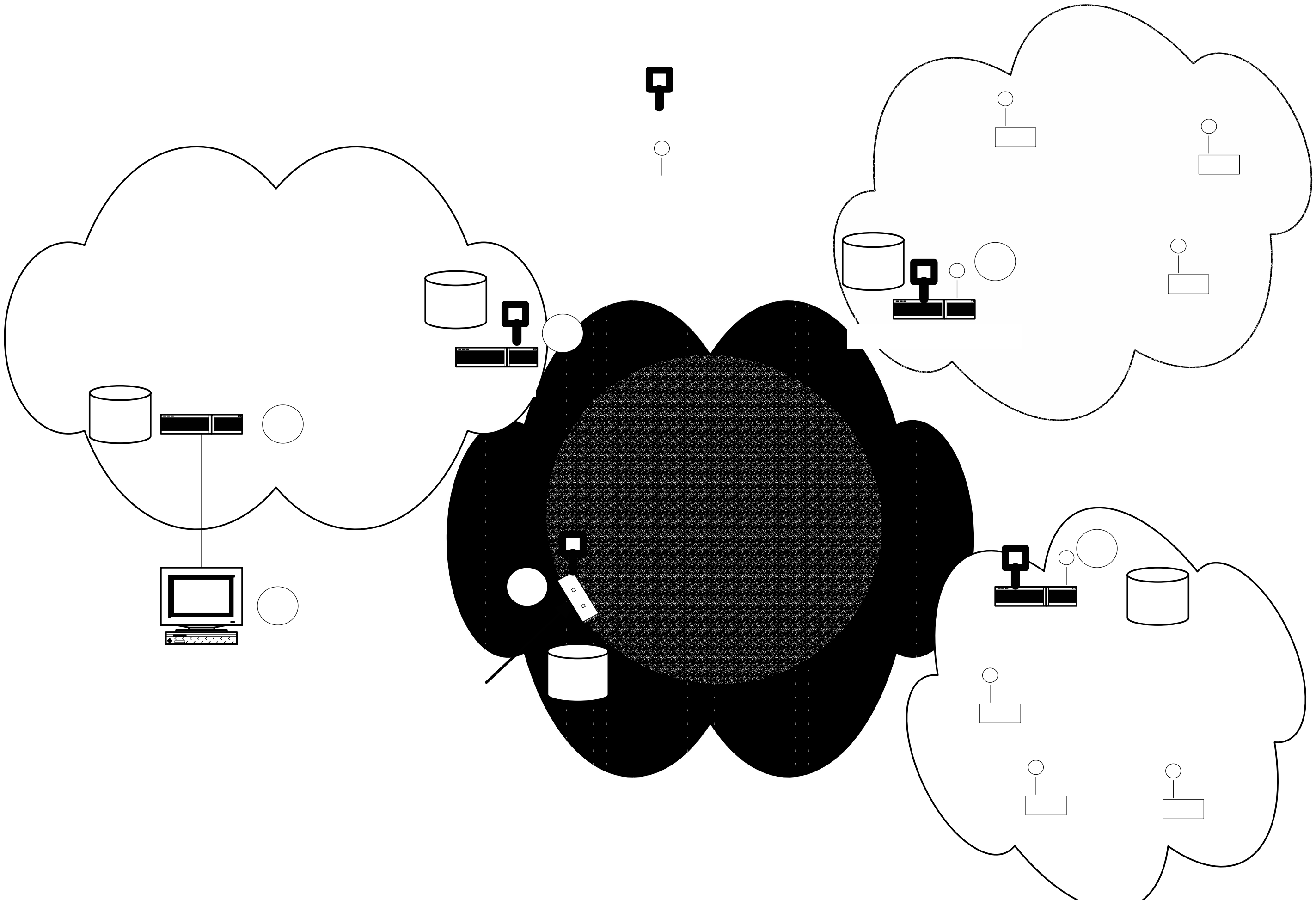
Naming

- Support 'radical heterogeneity' using *regions*:
  - Instance of an internet, not so radical inside a region
  - Common naming and protocol conventions
- Endpoint Name: ordered pair **`{R,L}`**
  - **`R`**: routing region [globally valid, topologically significant]
  - **`L`**: region-specific, opaque outside region **`R`**
- **Late binding** of **`L`** permits naming flexibility:
  - Associative or location-oriented names [URN vs URL]
  - May encompass esoteric routing [e.g. diffusion]
  - Perhaps an Internet-style URI [see RFC2396]
- *To do*: make **`R,L`** compressible in transit networks

Naming Challenges

- Structure of R (region name)
  - Variable length, hierarchical, centrally? allocated
  - Could likely use DNS namespace w/out mechanism
- How does a sender know/learn destination's R?
  - "just does" (like well-known port)
  - Some centralized or distributed service
- What semantic rules really apply to L?
  - Associative and location-based names seem useful
    - Associative – "send to Kevin's pager" [who looks up?]

Example Regions
(with Sensor Networks)



Reliable Message Overlay

- End-to-End Message Service: “Bundles”
  - “postal-like” message delivery over regional transports with coarse-grained CoS [4 classes]
  - Options: return receipt, “traceroute”-like function, alternative reply-to field, custody transfer
  - Supportable on nearly any type of network
- Applications send/receive bundles
  - “Application data units” of possibly-large size
  - May require framing above some transport protocols
  - Arrange for responses to be processed long after request was sent (application *re-animation*)

Routing on Dynamic Graphs

- Routing take place in time-varying topology
  - Links come and go, sometimes predictably
- Scheduled and Unscheduled Links
  - May be direction specific [e.g. ISP dialup]
  - May learn from history
- Link ``*Predictability continuum*''
  - S/U represents extreme cases regarding the expected availability of a route
  - Intermediate "predicted" category may evolve as a result of statistical estimation
  - Represent by a entropy-like measure (?)

Optimal Routing

- *Inputs*: topology graph, vertex buffer limits, contact set, prioritized message demand matrix
- A ***contact*** is an opportunity to communicate:
  - One-way: (t_s, t_e, S, D, C, D)
  - (t_s, t_e): contact start and end times
  - (S, D): source/destination ordered pair
  - C: capacity (rate; assume const over [*s*..*e*]); D: delay
- Vertices have buffer limits; edges in G if ever in any contact
- *Problem*: Compute the “best” set of paths for all messages so as to minimize total delivery time
- [formulated as LP – submitted to FOCS03]

Store and Forward

- Bundle routers generally have persistent storage
  - May offer *custody transfer* “service” if requested
  - Will try “very hard” to not discard messages for which it has accepted custody
  - Accepting custody for a bundle may involve a significant allocation of resources at a bundle router
- Some questions:
  - What do questions of flow and congestion control look like in one of these environment?
  - When should a bundle router avoid taking custody?
  - Given the hop-by-hop nature, if congestion control is figured out, does this also solve flow control?

Flow and Congestion Control

- Control at coarse time scales (“filesystem full”)
  - Very high delay → pre-schedule/admission control
  - Reasonable delay → dynamic flow control possible
  - Where does ‘traffic engineering’ end and ‘dynamic flow (congestion) control’ begin?
- For low-delay cases, which layer exerts FC?
  - Region-specific transports may support their own FC
  - Flow-control is logically hop-by-hop, so problem is to convert bundle-layer flow control to protocol-specific FC mechanism
  - Multiplexing multiple bundles on one transport causes problems due to head-of-line-blocking like phenomena

Some Security Issues

- Primary focus: *infrastructure protection*
  - Verify transit authorization at each overlay hop
  - Need some public-key facility for doing this
  - "Core" bundle routers must not be required to know every end-user set of credentials
    - Too big/slow; may be disconnected– difficult to look up
- Compromise for scalability
  - ACLs and user keys contained at firs-hop 'edge' routers
  - Edge routers authenticate and re-sign messages in their own keys
  - Next-hop routers need only check keys of its O(log n) [or maybe O(1)] neighbors

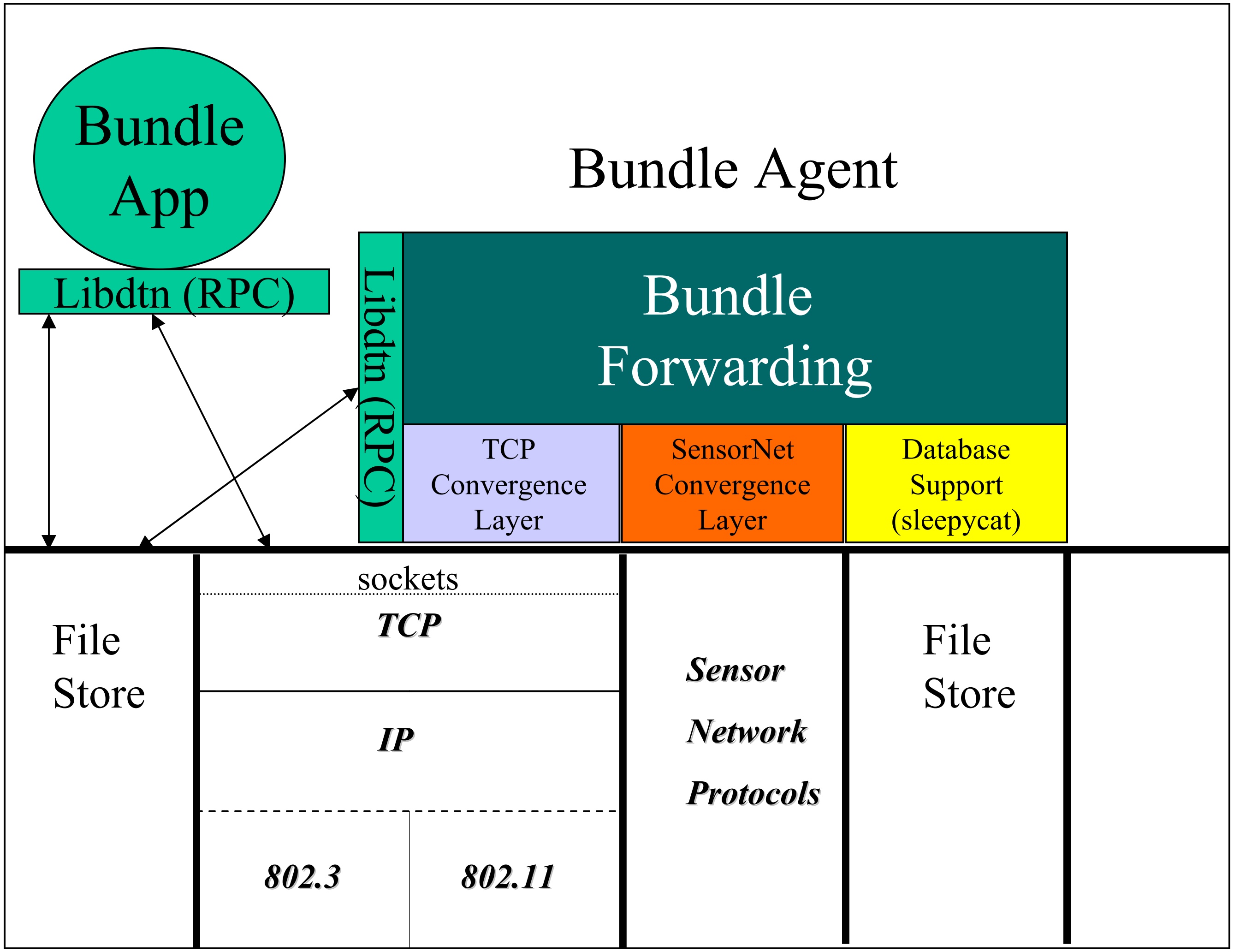
Security Issue Details

- Effect of a router compromise:
  - Router compromise could result in traffic being carried from that point onward
  - Router cannot completely masquerade as sender
    - Sending user still has its own private/public pair
- Compromise for scalability
  - ACLs and user keys contained at firs-hop ‘edge’ routers
  - Edge routers authenticate and re-sign messages in their own keys
  - Next-hop routers need only check keys of its O(log n) [or maybe O(1)] neighbors

Authentication of Fragments

- Consider xfer of bundle Z along link A->B
  - Z was signed by sender, but is also signed by A for transit through B
  - A->B link goes unavailable, but much of Z made it
- How to authenticate on fragments
  - Is there a keyed hash function that can take a substring (prefix) of a message and still somehow verify the signature [without using the ‘dice into chunks’ model]?

Bundle Forwarder

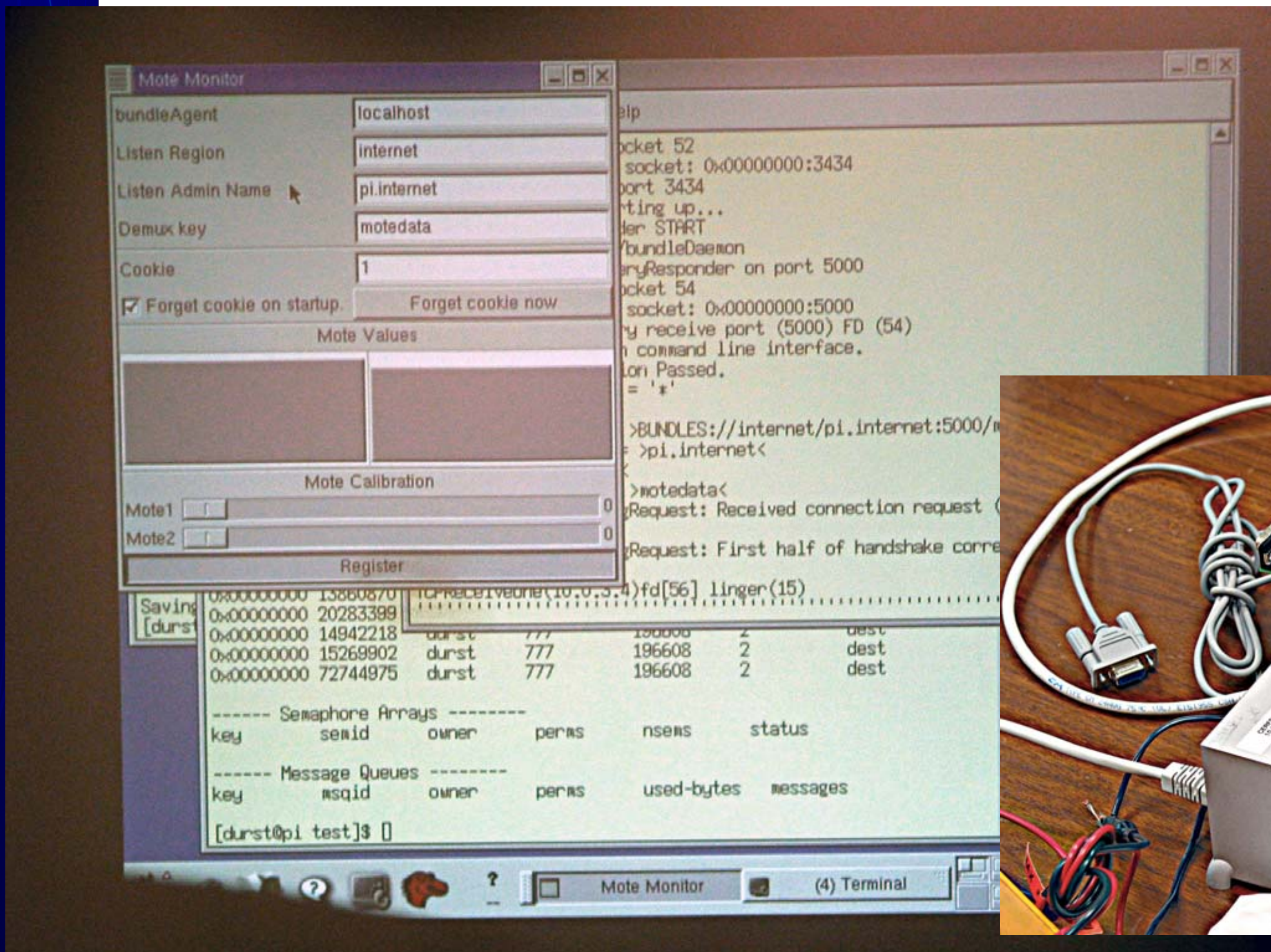


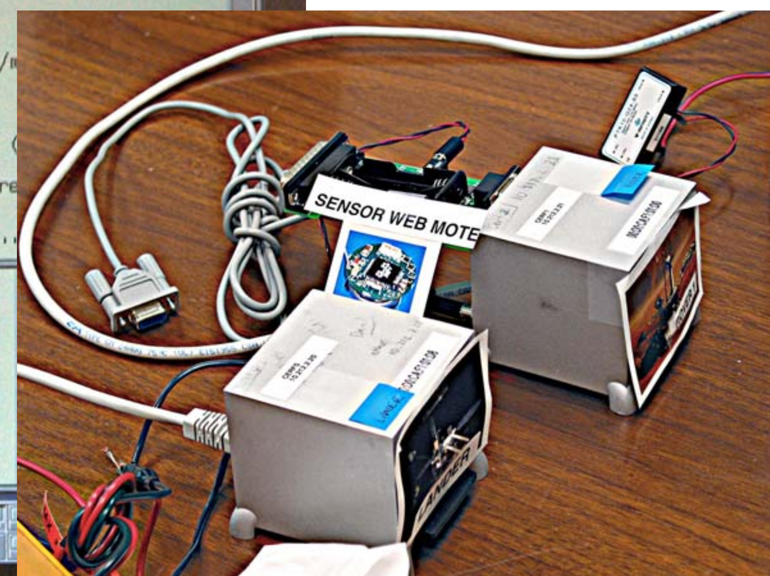
Application Interface

- RPC-based API is “split-phase” (libdtn)
  - RPC base allows for remote (dumb) clients
    - Apps are both clients and servers to RPC
  - sends decoupled from async receives
    - Request/response time may exceed longer than end-node lifetime
    - “Re-animation” capability to requestor or other
- Forwarder performs heavy lifting (bundledaemon)
  - Application (de)registrations
  - Executes convergence layers for send/receive
  - Bundle database maintenance
  - Basic routing functions

Demo (1)

Demo (2)


Mote Monitor
bundleAgent
localhost
Listen Region
internet
Listen Admin Name
pi.internet
Demux key
motedata
Cookie
1
Forget cookie on startup
Forget cookie now
Mote Values
Mote Calibration
Mote1
Mote2
Register
socket: 0x00000000:3434
rting up...
/bundleDaemon
eryResponder on port 5000
socket: 0x00000000:5000
y receive port (5000) FD (54)
n command line interface.
>BUNDLES://internet/pi.internet:5000/
>pi.internet<
>motedata<
Request: Received connection request
Request: First half of handshake corre
0x00000000 20283399
0x00000000 14942218
0x00000000 15269902 durst 777 196608 2 dest
0x00000000 72744975 durst 777 196608 2 dest
------ Semaphore Arrays --------
key semid owner perms nsems status
------ Message Queues --------
key msqid owner perms used-bytes messages
[durst@pi test]$
Mote Monitor
(4) Terminal


SENSOR WEB MOTE
LANDER

So, is this all just e-mail?

	naming/ late binding	routing	flow contrl	multi-app	security	reliable delivery	priority
e-mail	Y	N	Y	N	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

Status

- IETF/IRTF DTNRG formed end of 2002
  - See http://www.dtnrg.org
- DTN Agent Source code released 3/2003
- Several available documents (currently ID's):
  - DTNRG Architecture document
  - Bundle specification
  - Application of DTN in the IPN

Acknowledgements

- People (design & agent implementation):
  - Bob Durst, Keith Scott (MITRE)
  - Kevin Fall (Intel Research)
- More people (vision, design, commentary):
  - Vint Cerf (MCI)
  - Scott Burleigh, Adrian Hooke (NASA/JPL)
  - Juan Alonso (SICS)
  - Howard Weiss (SPARTA)
  - Forrest Warthman (Warththman)
  - Stephen Farrell (Ireland)
  - The *dtn-interest* mailing list

For more Information

- Delay Tolerant Networking Research Group
  - **http://www.dtnrg.org**
- Intel Research
  - **http://www.intel-research.net**
- IRTF Web Page:
  - **http://www.irtf.org**

kfall@intel-research.net

Thank you…