Delay-Tolerant Networking for Challenged Internets

Kevin Fall
Intel Research
Berkeley, CA
kfall@intel-research.net
http://www.intel-research.net

Unstated Internet Assumptions

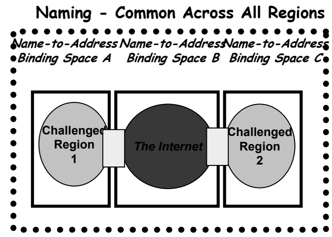
- End-to-end RTT is not terribly large
  – A few seconds at the most
  – (window-based flow/congestion control works)
- Some path exists between endpoints
  – Routing finds single "best" existing route
    - [ECMP is an exception]
- E2E Reliability using ARQ works well
  – True for low loss rates (under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

New challenges…

- Very Large Delays
  - Natural prop delay could be seconds to minutes
  - If disconnected, may be much longer
- Intermittent and Scheduled Links
  - Scheduled transfers can save power and limit congestion; scheduling required for rare link assets
- High Link Error Rates
  - RF, light or acoustic interference, LPI/LPD reasons
- Different 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 environments
- 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

- A *region*:
  - Instance of an internet
  - Common naming and protocol conventions
- Tuples (names): ordered pairs (R, L)
  - R: routing region [globally valid, topologically significant]
  - L: region-specific, opaque outside region R
- Late binding of L permits naming flexibility:
  - May encompass esoteric routing [e.g. diffusion]
  - Could be object names, addresses, queries, etc.
  - Relates to flexibility of URL suffixes
- Want to make L compressible in transit networks

Reliable Message Overlay

- End-to-End Message Service: “Bundles”
  - “postal-like” message delivery over regional transports
  - Optional reliability, class of service, return receipt, and “traceroute”-like function with alternative reply-to indicators
- Key Idea: Reliability via *Custody Transfer*
  - *Current Custodian* owns reliable-delivery guarantee
  - Bundles transferred between custodians toward destination
  - Sender may free resources upon successful custody transfer (destination considered an eligible custodian)

Message State

- Two distinct node types
  - P nodes: have persistent storage available
  - NP nodes: no persistent storage
  - P nodes might accept custody, NP nodes do not
- P node handling of custody transfers
  - Messages are stored persistently
  - Modifications to message forwarding state are treated as database operations (a database runs at P node message switches)
  - Forwarding engine replies with custody ACK to tuple indicated in the message “reply-to” field [sender may have to forward contents to this node for reliability]

Types of Routes

- Scheduled and Unscheduled
  - Scheduled: known ahead of time
  - Unscheduled: opportunistic contact
- S/U characterization is direction-specific
  - Consider the two ends of a user/ISP 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 (?)

The Routing Problem

- A *contact*:
  - Communication opportunity, parameterized as:

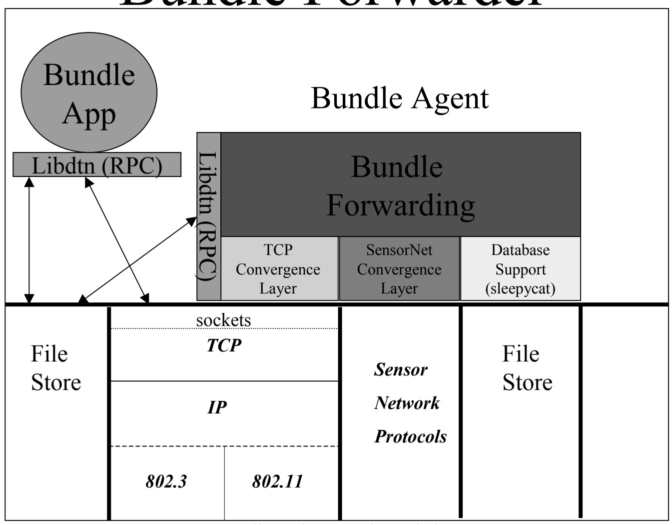
    (t_s, t_e, S, D, C, T)
  - (t_s, t_e): contact start and end times, if known
  - (S, D): source/destination pairs
  - C: contact capacity (rate); T: contact type
- A *message*:
  - Unit of transfer, parameterized as:

    (B, P)
  - B: message size (bytes); P: message prio [1..4]
- *Problem*: Compute "best" next hops for every message given a set of contacts [return to this…]

Flow Control

- Assume underlying protocols support some form of FC (either dynamic or static via a form of admission control)
- Flow-control is logically hop-by-hop, so problem is to convert flow control required at bundle layer to protocol-specific FC mechanism
- Fairly straightforward mapping problem when priorities are not included
  - With priorities, more sophistication required
  - In particular, how to map availability of (shared) buffers at bundle layer to protocol specific notions of flow control (e.g. slower reads on lesser prio TCP streams?)

Bundle Forwarder

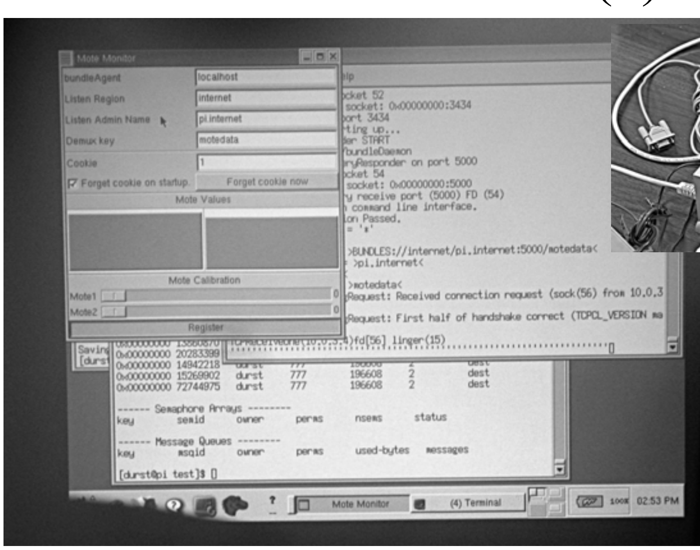


API Sketch

- Application API is "split-phase" using RPC
  - Message sends decoupled from async receives
  - Send message from memory or file
  - Establish handler for message receipt
    - → persistent: can cause "re-animation"
  - Apps may poll for arrived messages
- Current implementation is multi-threaded

Recent Demo (1)

Recent Demo (2)



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

- DTN work based on earlier IPN Architecture
  - IPN: Interplanetary Internet (www.ipnsig.org)
  - Developed notion of bundling and naming
  - DTN extends and generalizes IPN to non-space environments
  - IRTF IPNRG group produced arch draft (now expired)
- Prototype Implementation
  - ~15K lines of C code implementing DTN message switching prototype
  - Demonstrated support of Berkeley "motes" (sensors) and cfdp (JPL's file delivery protocol)

Futures

- Continue research and development
  - To implement: implement custody transfer, improve robustness of TCP convergence layer, restart on disconnect
  - To design: appropriate security mechanisms
  - To research: solution to routing problem, application of DTN in other unusual environments
- Form a community
  - Transition existing IPNRG in IRTF to a broadened DTNRG

Acknowledgements

- People (vision, design, implementation):
  - Bob Durst (MITRE)
  - Scott Burleigh (NASA/JPL)
  - Keith Scott (MITRE)
- More people (vision, design, commentary):
  - Vint Cerf (MCI)
  - Adrian Hooke (NASA/JPL)
  - Eric Travis (GST)
  - The *ipn-team* mailing list at JPL