Delay-Tolerant Applications

And some thoughts on deep space

Ed Birrane, JHU/APL

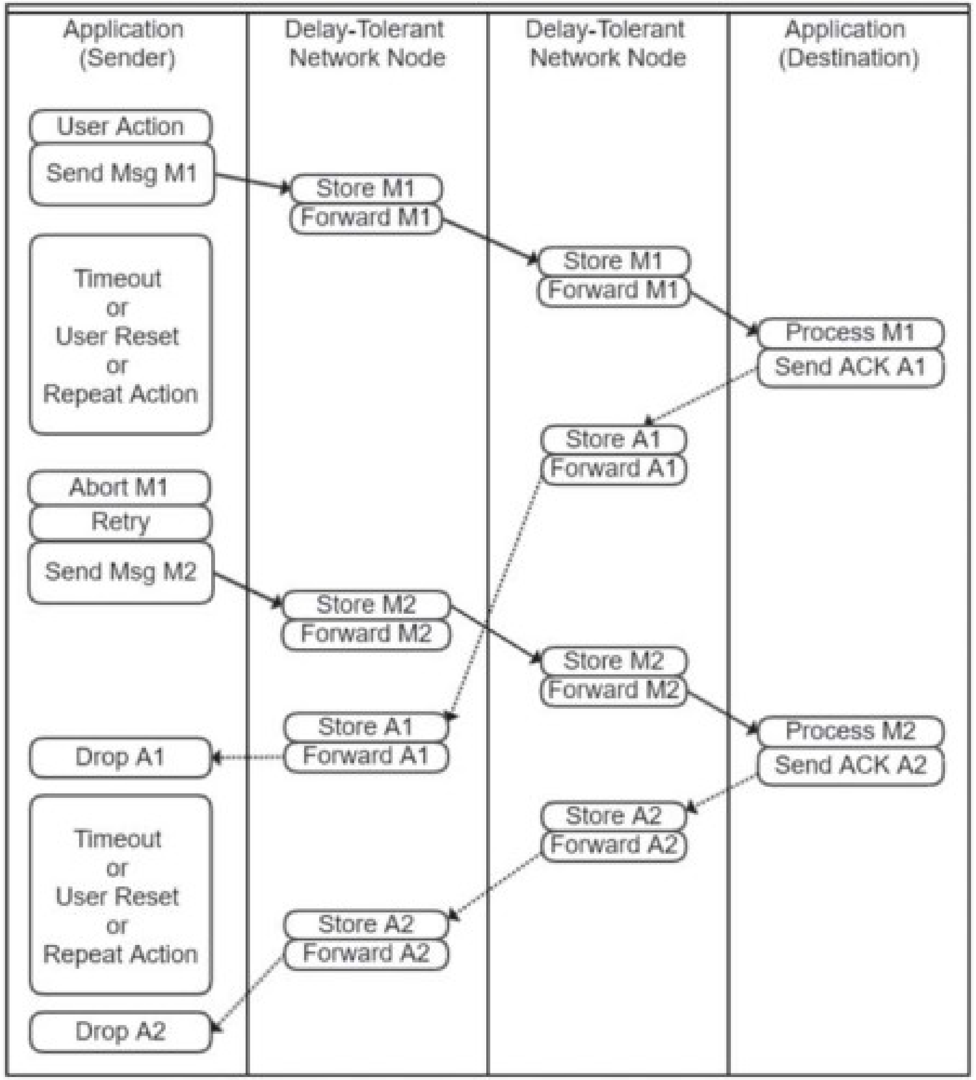
IETF-118

This presentation represents personal opinions only.

Chair hat is off.

Delay-Tolerance

- A tolerant network doesn't help an intolerant application.
- What are *some* things that make applications intolerant?
  - Using application timers
    - Timeout and Retransmission
  - Assuming state synchronization
  - Presuming plentiful local resources
- What helps?
  - Pre-place data
  - Push don't pull
  - Avoid sessions when possible



Birrane, Ed, and Jason Soloff. *Designing Delay-Tolerant Applications for Store-and-Forward Networks*. Artech House, 2020.

Latency vs Intermittency

- **Extreme latency**
  - Effective round-trip times are longer than various timers.
  - Assumes comms loss is due to timeout.
  - Can be solved by… extending timers.
- **Extreme Intermittency**
  - Endpoints in the network are removed from the network
  - Possibly because they rebooted or were turned off
- **Observations**
  - Extending timers does not help when “far” state is lost.
    - It might hurt…
  - Requiring even one RTT may be impossible
    - Consider Mars is up to ~40min RTT.
    - Radios might never stay on for 40 consecutive min.

A latency solution is not necessarily an intermittency solution.

Transport vs Communications Security

- Transport Security:
  - Securing the (a) transport layer
  - Assumes endpoint uses same transport layer
  - Assumes transport layers don't switch during transit
- Communications Security
  - Secure end-to-end information exchange
  - Property of the data itself
  - Independent of tunnels or transport layers
- Observation
  - End-to-end comsec should be independent of transport layer
  - Should include security-at-rest in a store-and-forward network

Assumptions

- Be careful when assuming **systems are capable and powered** because:
  - Telecommunications platforms may have short duty cycles
    - Yes, a NIC might always be on and powered, but a solar-powered radio is not.

- Be careful when assuming **long RTTs only affect timers** because:
  - Duty cycles might not span RTTs

- Be careful when assuming **local processing is sufficient** because
  - Maintaining multiple states/connections in RAM may not be possible.
  - Saving/restoring connections from persistent storage on reboot is complex and error prone

- Be careful when assuming **single secure transports** because:
  - Data may be waiting at endpoints or waypoints (secure at rest)
  - RTT-based negotiations can be problematic
  - Increasing RTTs to minutes/hours means the topology (and transport) might change

When to Consider BP-inclusive Stacks

- When you have intermittency not latency.
  - When your RTTs might exceed your duty cycles
  - When you cannot synchronize across an RTT

- When your end-to-end uses multiple transports
  - When your transport layer might collapse during exchange
  - When your transport layers might swap due to environment or conditions

- When your computer is constrained
  - Deep networking stacks and encapsulation
  - Slow CPUs and limited memory
  - Larger s/w images and longer boot times
  - Additional software processes, state, and OS features

Deep-Space Considerations (1/2)

- These comments focus on next say…5 years
  - IETF not IRTF
  - If something launches in 5 years it is being designed today.
    - Long lead times
    - High expense ($100m+ to $1b+)
  - Missions propose to a technology baseline
    - Not “we will invent it by the time we need it”
    - Moving off that baseline is difficult for operational missions
    - Tech demonstration missions exist to provide technical viability
- Examples
  - New Horizons ($700m) (12Mhz Mongoose with ~8MB RAM)
  - Parker Solar Probe ($1.3b) (80Mhz LEON3FT with ~22MB RAM)
  - DART ($330m) (~80Mhz LEON3FT UT700 with 16MB SRAM)



Deep-Space Considerations (2/2)

- Limited RAM
  - More RAM takes more power
- Limited CPU
  - Higher clock rates use more power.
  - Common to see 40-100Mhz processors.
- Low Duty Cycles (10-15%?)
  - Unnecessary components are kept off
  - Radios consume significant power
    - Transmit over very long distances
    - Contacts are usually planned

- Consider 40min RTT at Mars
  - Transmit data volume.
  - Turn off radio
  - Turn on radio ~40min later when responses expected.
- Typical CONOP
  - Charge batteries (majority of time. Hours. Days. Weeks.
  - Perform primary activities
  - Communicate only when necessary
  - **NOT**: Stay powered because someone else wants you to keep a session alive.

Power Constraints Dominate

Questions…

- Do we feel there is enough material for an informational RFC?
  - On how to write delay-tolerant applications?
  - On how to adapt protocols to run in DTN environments?
  - On the problems that define this area?

- Have we learned enough about DTN yet to:
  - Add to or update RFC4838?