Functional Analysis of I2RS: What Are We Putting in the Mixture?

Alia Atlas

IETF 86, Orlando, FL
Refining I2RS: the Feedback Loop

- Keep it reasonably scoped
- Don’t rush to the technology (but proto-typing could be useful)

• Use-cases justify and clarify the need for functionality
• New Functionality triggers ideas for new Use-Cases
It’s Just Like $foo Except...

• Trying to capture what is new/different/critical

• Concepts pulled from:
  – draft-ward-i2rs-framework-00
  – draft-atlas-i2rs-policy-framework-00
  – draft-rfernando-irs-framework-requirement-00
  – Other discussions
The Basics

• The **NEED** for **SPEED**
  – ability to react in sub-second time
  – Handled 100s, 1000s, etc. of operations

• Reasonable Programmatic Interface for Network Applications
  – Asynchronous
  – Not >5 different protocols to accomplish a thing
  – Pub/Sub model for Events
Multi-Headed Control (motivation)

• Multiple Clients (or Client + CLI) may want to write or modify the same state
  – CLI comes in to override client application state
  – Client application wants to override CLI (e.g. to enforce a policy)
  – **Use-Case:** Large-flow router and DoS mitigation both identify the same flow/destination to route.

• Depending on timing of processing of requests leads to *unknown* router state (which socket read first in a select, latency from an application)
Multi-Headed Control (implications)

• I2rs agent-based arbitration: enforce policy as to which client gets to modify contested state
  – Store if not best (like RIB policy)
• Client ownership: determines who can modify vs. delete/replace
• Notifications on subscription when state is changed by another client
• Optional garbage collection ephemeral state when client goes away
Different Operation Models

• Three characteristics to an operation:
  – Start-Time (immediate, temporal, triggered)
  – Persistence across reboots (permanent, ephemeral)
  – Duration/State-expiration (unbounded, temporal)

• CLI and Netconf generally give:
  Immediate, Permanent, Unbounded
Subscriptions and Notifications

• To be responsive to uncontrollable events, network applications learn via notifications
  – Has state been overwritten?
  – Has a next-hop changed?
  – Has a threshold been passed?
  – Has a route been installed?

• Ability to filter and specify thresholds per subscription request.

• Events and data-publication streams
Multiple Transport Sessions

• Different communications have different requirements
  – Reliability, secrecy, replay, etc.

• Clients and agents part of distributed systems
  – A network application may be distributed to different locations.
  – A network element may provide services via different system elements, want to distribute notifications and analytics from where they are learn or measured
Role-Based Client Identity

- An application can identify as a client even with:
  - Different IP address
  - Different TCP session
  - Etc.

- Supports distributed applications, standby for applications, etc.
Client Limited Authorization

- Ability to limit client ability to read and write based upon role.
  - Can learn subset of topology
  - Define write-scope in terms of values/ranges as well as data objects in model
Client-Specific Priority

- Client can specify a priority for each operation
  - Allows in-flight and ASAP operations
  - Operations across multiple channels can be put into desired order
Standard Information and Data Models

• RIB interactions:
  – Static routes, redistribution into other protocols, varying admin-distance
  – Variety of next-hop types
  – Unicast, multicast, LFIB, etc.

• BGP policy:
  – Actions, etc.

• IGP local policy

• PIM local policy
Topology

• Varieties of Data
  – Active IGP topology
  – Active components (e.g. customer or peer links)
  – Passive components (can be detected)
  – Passive components (undetectable)
  – Minimal historical (last up, last peer, etc.)

• Network Abstractions
  – Service points-of-presence
  – VPN topologies
  – unsummarized topology
Motherhood and Apple-Pie: Pick Your Recipe

- Atomic Operations
- Preemptable Locking: take ownership at precedence for blocking changes or to write
- Rollback by client
- Capabilities
- Optional garbage collection
Flexible Components to Consider

Data Modeling Language

YANG, ASN.1

NetConf, XMPP

XML, Google ProtoBufs, Binary

Transfer Syntax

Protocol

Transfer

Transport

TCP, UDP, TLS, SCTP
What to keep?

• Some functionality drives and determines the architecture...
• Some can be added later...
• Need to justify with use-cases and consensus

TIME to DISCUSS