Questions from the Security Review of Architecture document (draft-ietf-i2rs-architecture-07)

Sue Hares

Topics

- Questions from the security review
- My understanding of answers and open changes
- Review of where we are with ephemeral State
- Discussion about the Architecture view of the protocol

Questions

1. Is a new protocol needed?
2. Have you considered validating the secondary identity
3. Is design for security and the nature of the asynchronous protocol complete in this document?

4 reboot

a. Is there a difference between I2RS Client reboot and I2RS Agent reboot?
b. If the I2RS Client fails, does the I2RS Agent know how to identity an alternative one?

Questions

5. What is collision?
   - Å What happens If two requests at same time from different clients?
   - Å What happens if the a second client over-writes the first client.
   - Å How does this interact with role?

Question 1 – Do we need New protocol?

- My first question when I started reading this document was why do we need a new protocol.
  - Wouldn't SNMP or NETCONF do this just fine? And there are probably lots of others [*protocols]*
  - Section 3.1 says "There have been many efforts over the years to improve the access to the information available to the routing and forwarding system." It would be good to understand why those efforts failed before inventing some new syntax (when it is unlikely the syntax is what killed previous efforts).
  - Then section 7.1 says the protocol will be "based on" NETCONF and RESTCONF. What does "based on" mean in this context?

Q1: Is a new Protocol Needed?

- Why Did other protocols fail?
  - SNMP – MIBs hard to write, not programmatic
  - Light weight protocols: Apache THRIFT possible
  - NETCONF/RESTCONF – still in progress, but developers willing to work with us
- WG Direction -
  - All Feature Protocol: NETCONF and RESTCONF
    - Write draft defining additional features
    - Get input from implementers
  - Light-weight: Waiting for a proposal

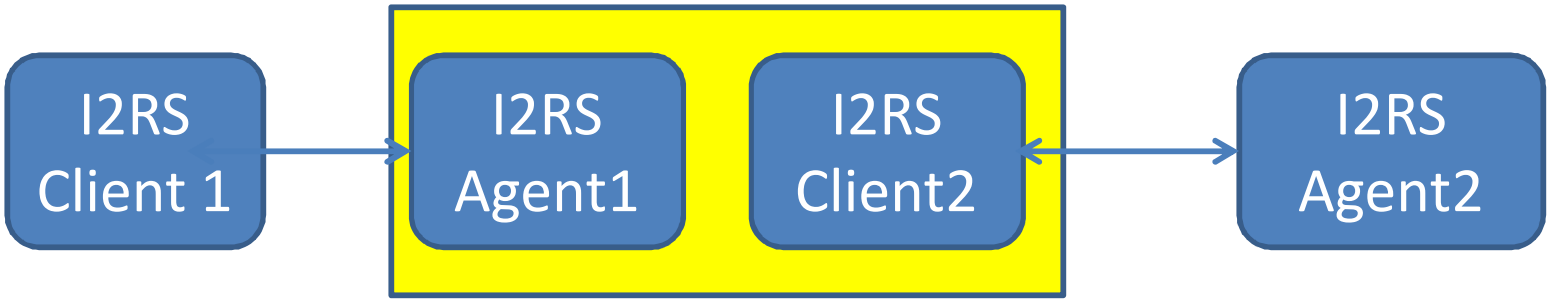
Q1: Next steps

- Requested presentations on protocol
  - RESTCONF features
  - draft-voit-i2rs-pub-sub-requirements-00
  - Any light-weight proposals
- Next decisions
  - DOES RESTCONF or RESTCONF+pub/sub meet I2RS's needs?
  - Should we support a Light-weight protocol + heavy weight?

Q2 – Validating credential of Second ID

Architecture reference: Secondary identity - forwarding some opaque identifier of that requesting entity to the I2RS Agent for logging.

- Use Case: Broker



Client 2 can pass “client1” as opaque string from client 2 To Agent 2

Q2 – Broker (Security Reviewer)

- **Section 4 par 3.** (and in some other places), you talk about the I2RS Client acting as a **broker** forwarding requests for some other entity, and forwarding some **opaque identifier** of that requesting entity to the I2RS Agent for logging.
  - This presumes that the I2RS is configured with (or has access to) the authorization information that says which requestors are permitted to do which operations.

- **A useful extension to the protocol** would be to be able to forward a requestor-identity string that the Agent not only logs but also checks for proper authorization before performing the requested operation.
  - The Agent would need to verify that both the Client and the client asserted identity of the requestor be authorized to perform the operation.
  - This relatively simple change to the Agent and the protocol might permit a considerably simpler client (if this brokered-request behavior is actually common).

Q2 WG Decisions

- WG Decision
  - Secondary identity is just opaque string which is passed without verification for purposes of trouble shooting and logging changes
  - After deploying this feature, if something else is needed we can revise the protocol.
- Next Step
  - Nothing as WG LC has completed
  - IETF LC is coming so let mail list or WG chairs if you have any concerns.

Q3.1: Security and Asynchronous Nature of I2RS protocol

Question: Is security and the support for the asynchronous nature of the protocol fully supported?

- Asynchronous usually means that you can launch operations and then check back later whether they successfully completed.
- If you want to execute a second operation only if a first succeeds (or to guarantee the order in which they execute), you need to at some point wait for operations to complete
- There is also substantial overhead in supporting asynchronous operation in that all transactions need labels so that they can be queried?
- **Have you done this?**

Q3.1 Answers

- **WG Decision:**
  - **I2RS client** must act and check back to see if state change.
  - Collision of Simultaneous Writes is considered error.
  - Last Write wins is the current status
  - I2RS Architecture current does not have label or checkpoint.
- **Checkpoints or transaction labels**
  - RESTCONF : No Checkpoint or transaction Labels exists in RESTCONF.
  - RESTCONF session is an atomic action
  - NETCONF writes to Candidate store and can roll-forward (Commit) or roll-back.

Q3.1 Next Steps

- **Decision 1:** Does I2RS protocol require a checkpoint?
  - Checkpoint == 1 monotonically incrementing counter with discontinuity
- **Decision 2:** Does I2RS require Transaction labels?
  - WG decision: No
- Will send a note to list.

Q3.2 Security and Asynchronous Nature of I2RS protocol

- **A conceptually simpler strategy** is to say that since a client can make multiple parallel connections to an agent that in cases where a client wants asynchronous operation he opens multiple connections and launches one asynchronous operation on each.
  - The cost is that is has lower performance in cases where there are large numbers of parallel operations tying up lots of connection state.

Q3.2 Asynchronous Nature

- **WG status:**
  - WG has approved: Multiple parallel connections.
  - WG has approved: RESTCONF as a candidate protocol, and RESTCONF has 1 asynchronous set of operations per connection.
- **Decision point:**
  - 1) Only multiple parallel sessions with 1 set of asynchronous operations per node (RESTCONF).
  - 2) Transactions with Roll-forward/Roll-Back at Commit (NETFCONF),
  - 3) Full Transaction mode.
  - Or some combination 1, 2, or 3
- **Current architecture:**
  - 1 + 2 + 3 allowed (RESTCONF, NETCONF, and something else).
- **Q:** Should we restrict to 1 + 2?

Q3.3 Security and Asynchronous Nature of I2RS protocol

- **Comment:** The last paragraph of 7.9 says "the protocol will include an explicit reply to modification or write operations even when they fully succeed". How does this relate to the asynchronous nature of the protocol?
- Response
  - WG Past Decision: Protocol would include explicit responses to change or writes.
- **WG Status:**
  - RESTCONF - appears to provide this feature
  - NETCONF - ???
- **WG decision**: Is there a problem with replies?

Topic 4 - Ephemeral State in Reboot

- **Ephemeral State.**

a. Is there a difference between I2RS Client reboot and I2RS Agent reboot?

b. If the I2RS Client fails, does the I2RS Agent know how to identity an alternative one?

- **Problem**
  - Some devices do not have a concept of ephemeral state

Topic 4.1: Ephemeral State in Reboot (1)
Security person

- **Section 6.2:** The restriction that this protocol injects only ephemeral state seems surprising, especially given that the circumstances under which the ephemeral state is lost are defined in terms of a network device reboot.
  - **Some network devices may not have a clear notion of a reboot**, or might do it so rarely as to render such functionality useless. I was confused by the discussion of agent reboots vs. device reboots.
- **Text issues: 6.2**
  - The first paragraph seems to say that ephemeral state is lost when the device reboots, but 6.2.1 seems to imply that state is lost when the agent reboots.
  - The sentence "Just as routing state will usually be removed shortly after the failure is detected…" seems to imply that ephemeral state might be lost when a client reboots.
- **Text 6.3 Have you considered what happens to state when a client disappears but the agent and server stay around forever.**
  - There is an option later in the document for some sort of timeout, but I would think there would be some sort of mechanism to guarantee that all ephemeral state disappears eventually unless the requestor is still around implicitly renewing it.

Topic 4.1 Ephemeral State in Reboot (2) (security reviewer)

- Also in 6.2.1, it appears that one piece of state is explicitly not ephemeral... the agent keeps a non-ephemeral list of clients to notify when ephemeral state is lost.
  - If the client is not accessible, for how long does the agent continue to try to contact it? Forever?
- **The protocol requires that agents be able to open connections to clients** (in addition to clients being able to open connections to agents).
  - This will introduce lots of challenges. It means the client needs an open port to accept connections, likely an SSL certificate, and will be in trouble if it is behind a NAT or is mobile and does not have a stable IP address.
  - **Other parts of the spec mention that two entities** might have the same client identity. In such cases, it will be tricky for the agent to connect to "the right instance".
  - It might be better to only allow clients **to initiate connections to agents**, possibly with some sort of unauthenticated notification from agent to client that initiating such a connection would be a good idea (to reduce the overhead of the polling that would otherwise be necessary).

Topic 4 - Ephemeral State in Reboot

Å **WG Ephemeral State.**

a. Differences I2RS Client reboot and I2RS Agent reboot are different, but the differences are handled by the i2rs Architecture descriptions.
b. I2RS Client fails, does the I2RS Agent knows how to reach another one for notifications or publications.

Topic 4 - Details

- **Current WG status on Agent**
  - **A**gent reboot and client reboot in ***some (not all)*** contexts.
  - Agent reboot may be due to system or agent software.
  - Agent which reboots loses ephemeral state and connections state, but has NVM on which clients were connected to.
  - Agent sends clients “rebooted” messages if previous connected
  - Agent send clients request to connect if needs to publish or notify.

- **Current WG status on Client**
  - Client reboots may or may not lose state,
  - Timeouts on client side of connection depend on application
  - draft-voit-i2rs-pub-sub-requirements-00 – allows pub/sub to be sent else where
- **Question :** Does architecture miss anything?

Topic 5 - Collisions

- **Collisions**
  - Text unclear if these were collisions in the time sense where two requests are made simultaneously by different clients vs. whether it is a case where once client tries to override the setting of another client.
- **Problem with question brings up**
  - The role should prevent the write of one client over another client unless both have write roles.
  - Should there be only 1 write role per node; and is this a agent issue
- **Current WG status:**
  - Two Writes is an error, Client should handle.
  - Two writes, latest one wins.
- **Decision:** Should we change?

Q5: Collisions (Security person)

- Section 7.8 talks about "collisions",
  - but it wasn't clear (at least to me) whether these were collisions in the time sense where two requests are made simultaneously by different clients vs. whether it is a case where once client tries to override the setting of another client.
- Two clients validly update same parameter
  - I also wonder whether there are cases where two changes would interact in some way other than one of them winning, as when two clients each want to increment the bandwidth of some virtual like over which they are both tunneling traffic (and where the correct result is to add the two increments).

Q5: Collision

- Q5: Decision:
  - Collisions are an error that clients must prevent
  - Clients overwriting each other sequentially – is a client problem
    - Here: a checkpoint + discontinuity has been discussed
    - No decision
  - Decided the combination of two clients is the client issues.
- Next Steps:
  - Do these decision cover the Qs?
  - If not, what should we add.

Summary

1. (Q1) DOES RESTCONF or RESTCONF+pub/sub meet I2RS's needs?
2. (Q1) Should we support a Light-weight protocol + heavy weight?
3. (Q3.1) Does I2RS protocol require a checkpoint?
4. (Q3.1) Does I2RS require Transaction labels?

Summary of Decisions (2)

5. Q3.2 Decision point:

- 1) Only multiple parallel sessions with 1 set of asynchronous operations per node (RESTCONF).
- 2) Transactions with Roll-forward/Roll-Back at Commit (NETFCONF),
- 3) Full Transaction mode.
- Or some combination 1, 2, or 3

Å **Current architecture:**

- **Current we allow any combination of 1, 2, or 3**

Å **Question:** Should we restrict to 1 + 2?

Confirm these decisions

- **(Q2) –** Secondary ID is just opaque string
- **(Q3.1)** No transaction labels required in protocol
- **(Q3.3)** WG decision: Replies on Writes (e.g. in RIB-Info)

- **(Q4) Ephemeral State**
  a. Differences I2RS Client reboot and I2RS Agent reboot are different, but the differences are handled by the i2rs Architecture descriptions.
  b. I2RS Client fails, does the I2RS Agent knows how to reach another one for notifications or publications.

- Q5: **Collision**
  - Two Writes is an error, Client should handle.
  - Two writes, latest one wins.