I2rs Requirements for NETCONF

Susan Hares (i2RS Co-chair)

I2RS Requirement on WG LC

- draft-ietf-i2rs-ephemeral-state-00
- draft-ietf-i2rs-pub-sub-requirements/
- draft-ietf-i2rs-traceability/
- draft-ietf-i2rs-protocol-security-requirements-01

Adopted environmental security requirements

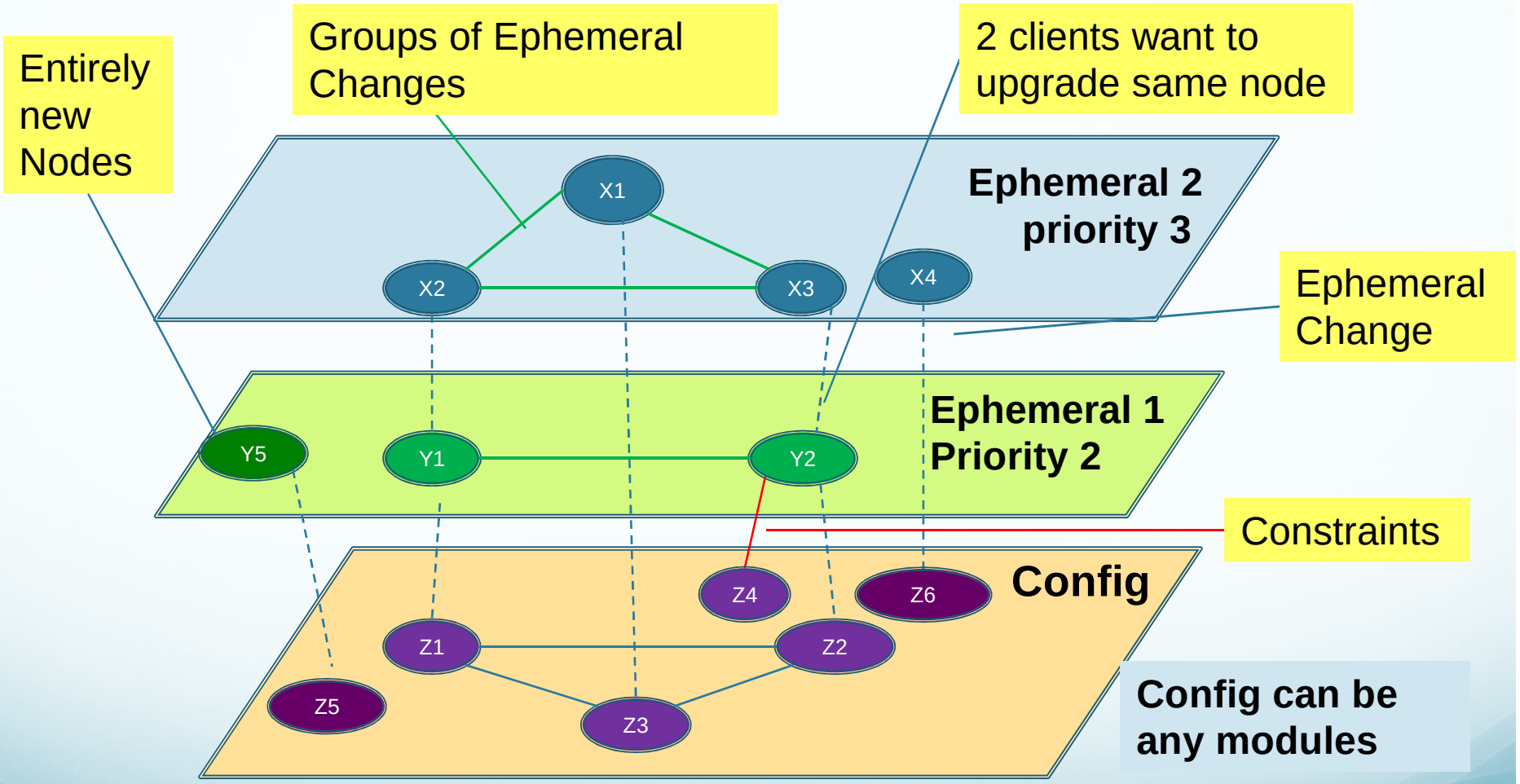
- draft-mglt-i2rs-security-environment-reqs-01

Summary

- Ephemeral state from the protocol strawman
- Yang changes
- Identity requirements
- Priority requirements
- Pub/sub requirements
- Security requirements

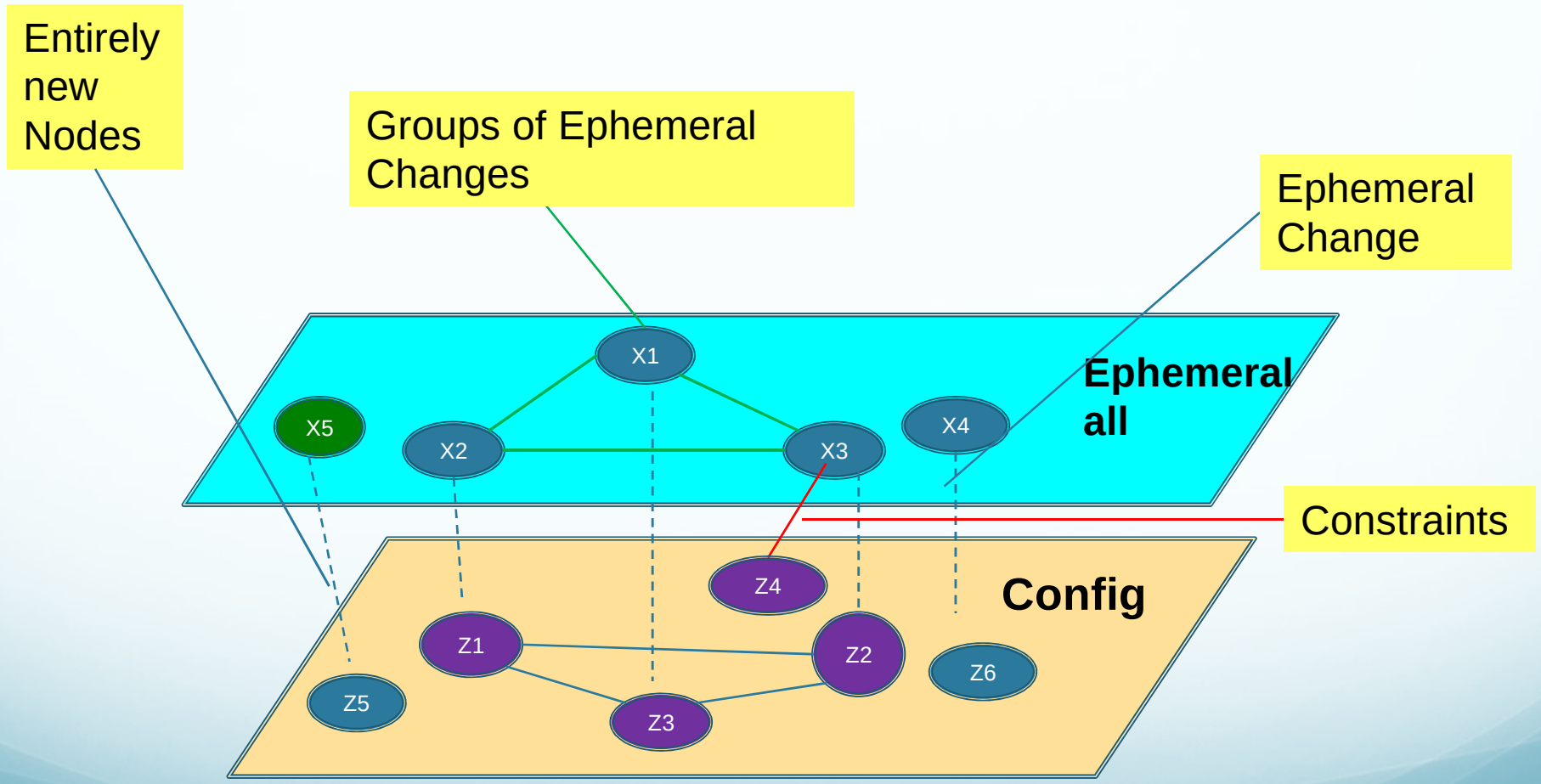
Ephemeral State

Panes of Glass Model



“N” panes of glass allows caching
Handling of non-linked as orthongonal writes
(aka - stop on error, continue on error)

Simplified Panes of Glass Model



2 Panes of Glass – all or nothing

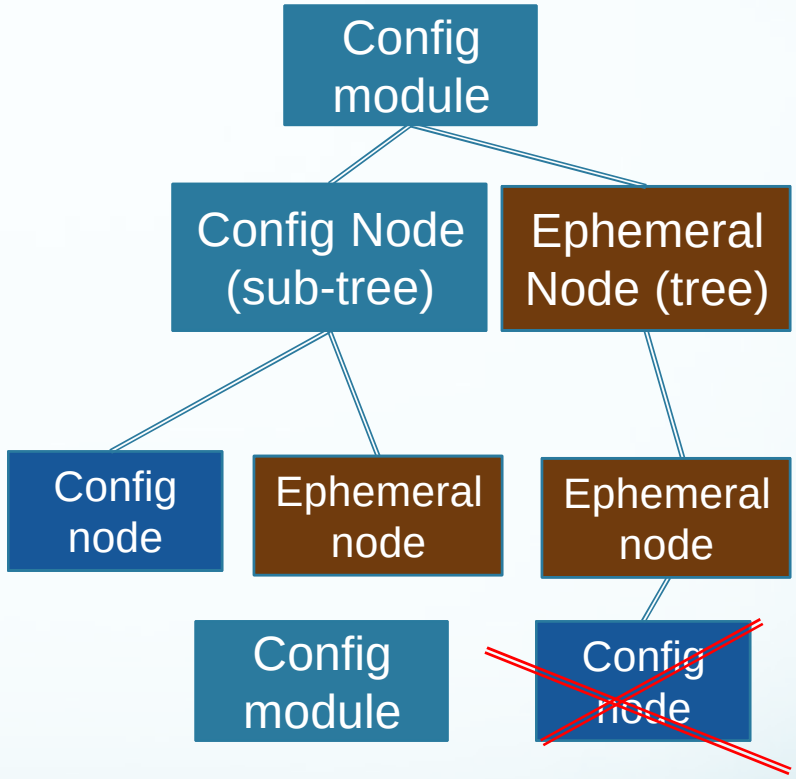
Ephemeral State

1. **Ephemeral state is not unique to I2RS**
2. **The ephemeral data store is a data store holds configuration that is intended to not survive a reboot.**
3. **Ephemeral state can be in any data model – so importance of ephemeral is for conformance checking**
4. **Ephemeral data store is never locked**
5. **Ephemeral data store can occur in two ways:**
   1. **Yang module that contains both non-ephemeral and ephemeral**
   2. **Yang module that only contains non-ephemeral**

The yang modules may be protocol modules (BGP) or protocol independent modules (RIB, FB-RIB, Topology)

Ephemeral State (2)

6. Ephemeral nodes may not have configuration nodes beneath
   a. Ephemeral modules may not have configuration within the module.
   b. Configuration modules may have ephemeral Nodes and configuration modules



Ephemeral State (3)

7. Ephemeral state will be denoted by “ephemeral” in Yang protocol at node level, submodule, or module level

```
module thermostat {
    …
    leaf desired-temp {
        type int32;
        ephemeral true;
        units “degrees Celsius”;
        description “The desired temperature”;
    }

    leaf actual-temp {
        type int32;
        config false;
        units “degrees Celsius”;
        description “The measured temperature”;
    }
}
```

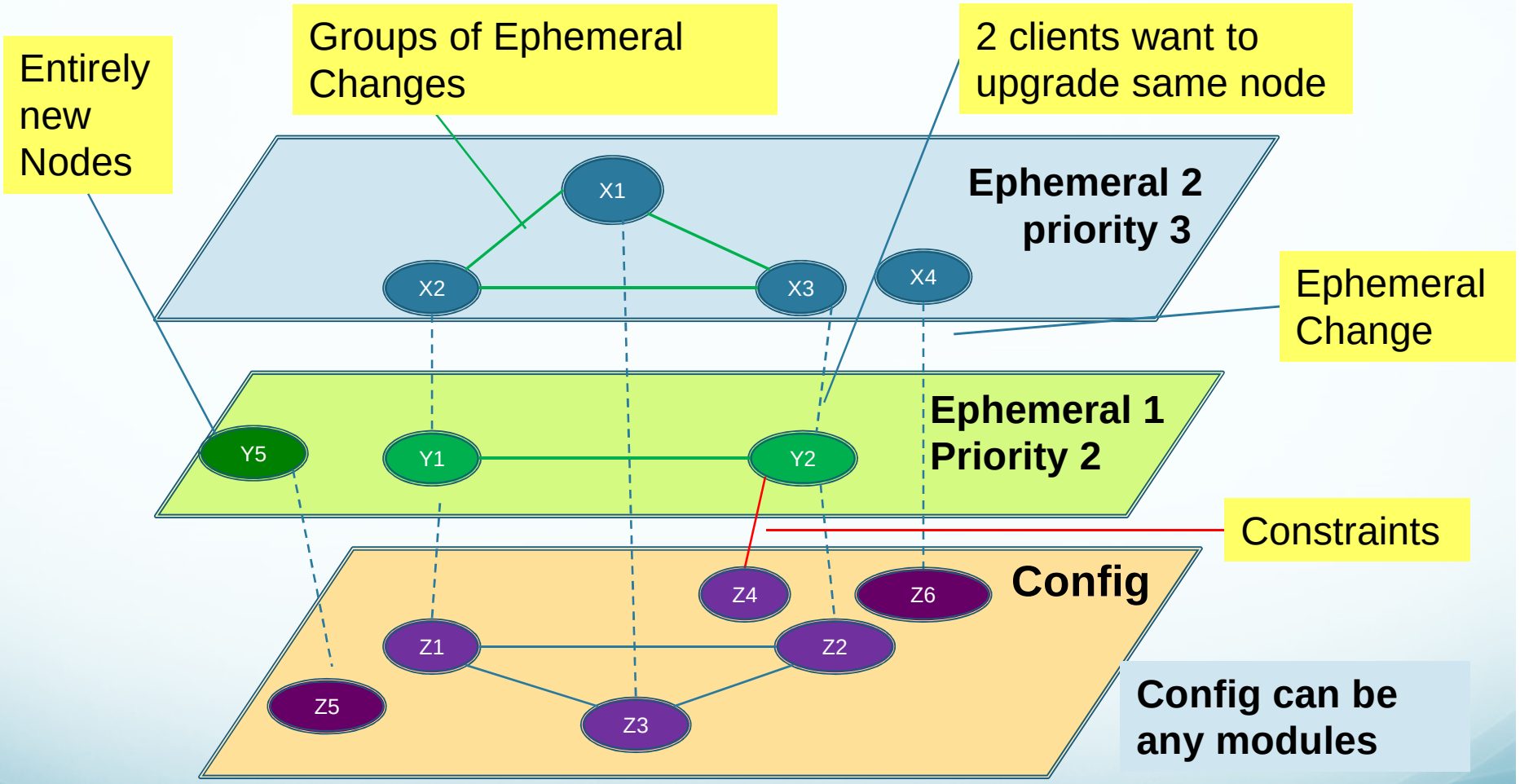
Ephemeral State (4)

8. Ephemeral has two error handling extensions
   1. Ephemeral data store allows for reduced error handling that MAY remove the requirements for leafref checking, MUST clauses, and instance identifier (to allow more speed)

   2. Ephemeral data store allows for priority resolution of write operation
      - Priority error resolution means each I2RS client of the ephemeral I2RS agent (netconf server) **MUST BE** associated with a priority.
      - Priority write resolution occurs when a I2RS client with a higher priority writes a node which has been written by an I2RS client (with the lower priority).
      - When the I2RS agent (netconf server) allows a higher priority client to overwrite a lower priority client, the I2RS Agent **MAY** provide a notification indication to entities monitoring the node.

Should MAY be MUST?

Panes of Glass Model



“N” panes of glass allows caching
Handling of non-linked as orthongonal writes
(aka - stop on error, continue on error)

Example of Client nodes

```
container i2rs-clients {
     leaf max-clients {
        config false;
        mandatory true;
         type uint32 {
            range "1 .. max";
          }
    }

   list i2rs-client {
          key name;
          unique priority;
          leaf name { ... }
         leaf priority { ... }
    }
}
```

Ephemeral State (5)

9. Caching – is out of scope for the first I2RS protocol release.

- Long-term concern: latency of I2RS protocol

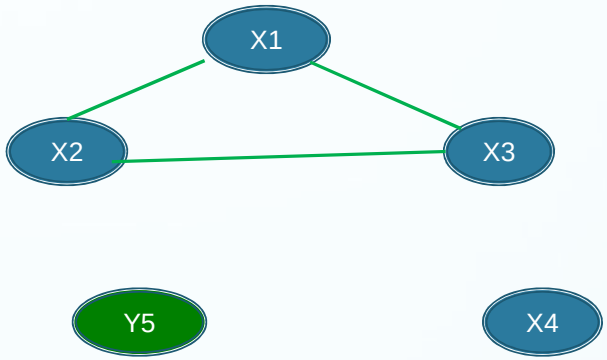
Error handling

Types of error checking

- Syntax - correct syntax for node
- Referential – leafref, MUST, instance identifier
- Grouping – group of nodes that should align

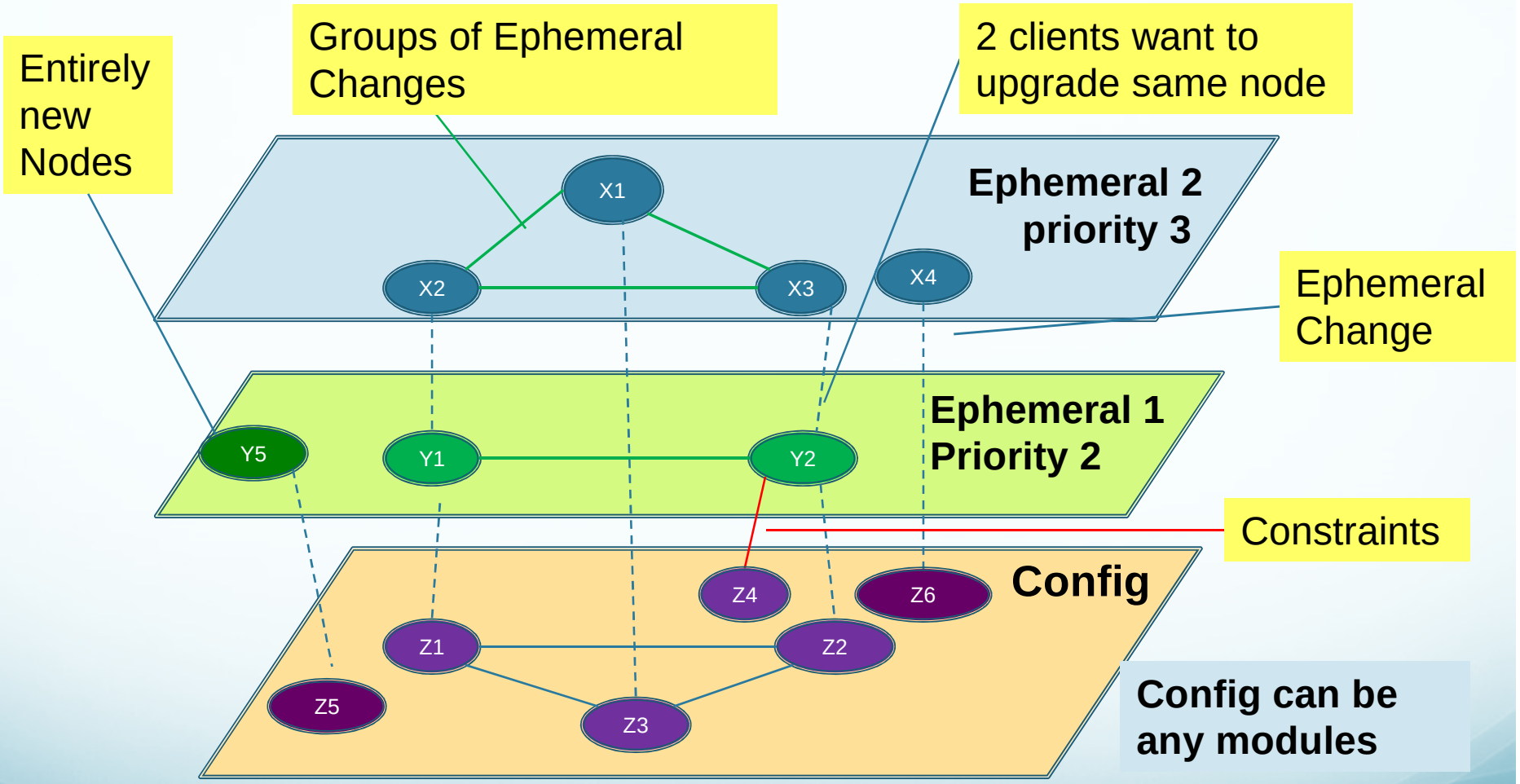
Error handling

- Two types of data
  - Grouped data - data must be done together
    - Examples:
      - BGP static routes + cost communities
      - Interface addition + routes
  - Non-Grouped data
    - Example: PCEP state + BGP state



- "Stop-on-error" and "continue-on-error"
  Assume non-grouped data

Panes of Glass Model



“N” panes of glass allows caching
Handling of non-linked as orthongonal writes
(aka - stop on error, continue on error)

Examples of Error handling

Actions that might ignore referential checks

- RIB routes added for DOS
  - I2RS Client may be able checks on routes
  - I2RS Agent could remove referential checks
  - 100K routes would go faster
  - May not have valid
- BGP Cost communities added
  - I2RS Client may be able to checks

Actions that should not ignore referential checks

- BGP Peer added ephemeral state
  - BGP peer needs checking to be valid

Summary: Model dependent ability to ignore referential checks

Support for Partial Writes

- NETCONF
  - No mandated sequencing of edit functions or write functions
  - Without mandated sequences, NETCONF can not handled partial edits
- RESTCONF
  - Complete set of operation per message
  - RESTCONF can support multiple data messages

I2RS Architecture

Error handling:

- **stop-on-error** - means that the configuration process stops when a write to the configuration detects an error due to write conflict.
- **continue-on-error** - means the configuration process continues when a write to the configuration detects an error due to write process, and error reports are transmitted back to the client writing the error.
- **all-or-nothing** - means that all of the configuration process is correctly applied or no configuration process is applied.

Initial I2RS Protocol: "**all-or-nothing**" in I2RS Agent

Yang changes

Yang changes

- “ephemeral” in Yang protocol at node level, submodule, or module level

```
module thermostat {
    …
    leaf desired-temp {
        type int32;
        ephemeral true;
        units “degrees Celsius”;
        description “The desired temperature”;
    }

Operational state – not supported (???)

    leaf actual-temp {
        type int32;
        config false;
        units “degrees Celsius”;
        description “The measured temperature”;
    }
}
```

Identity, Priority, Capabilities

Identity Requirement

- Clients and I2RS agents shall have identities
- Clients can have secondary identities
  - Carried as part of ?? (RPC, Meta-data) –still open in protocol design, but it is just an opaque value
  - Goes in as part of the notification/trace log

Priority Requirements

- Support multi-headed control –
  - Different than ephemeral
  - Supported as part of write-collision
- I2rs attributes may be modeled as meta-data

Signaling capabilities

- Yang library supports
  - Modules or submodule ephemeral
  - Data schema indicates nodes

```
+--ro modules
    +--ro module*[name revision]
    +--ro name yang: yang-identifier
    +--ro revision union;
    +--ro schema? inet:uri
    +--ro namespace inet:uri
    +--ro feature* yang:yang-identifier
    +--ro ephemeral
    +--ro deviation* [name revision]
    |   +-- ro name yang:yang-identifier
    |   +-- ro revision union
    +--ro conformance enumeration
    +--ro submodules
        +--ro submodule*[name revision]
            +--ro name yang:yang-identifier
                +--ro revision union
                +--ro schema? inet:uri
                +--ro ephemeral
```

Security Requirements

I2rs Security Requirements

- Requirements 1, 2, 5, 6, 7, 9, 11, 13, 14, 15, 16, 18, 19, 20 (OK)
- Edited requirements: 3,4, 10 – OK
- Security examined reviewed
  - 8,12 (DDoS) – OK
  - multiple messages – remove
  - insecure protocol
  - Insecure protocol
    - Only if Data model clearly states it.
    - Call for RIB to determine if can state it.

Secure protocol fits

- NETCONF
  - SSH - RFC6242
  - TLX with X.509 – RFC7589
- RESTCONF
  - XML or JSON with RFC7168
  - http with : GET, POST, PUT, PATCH, DELETE, OPTIONS, HEAD
- Exception – pub/sub features and traceability features already proposed in NETCONF

Insecure Protocol

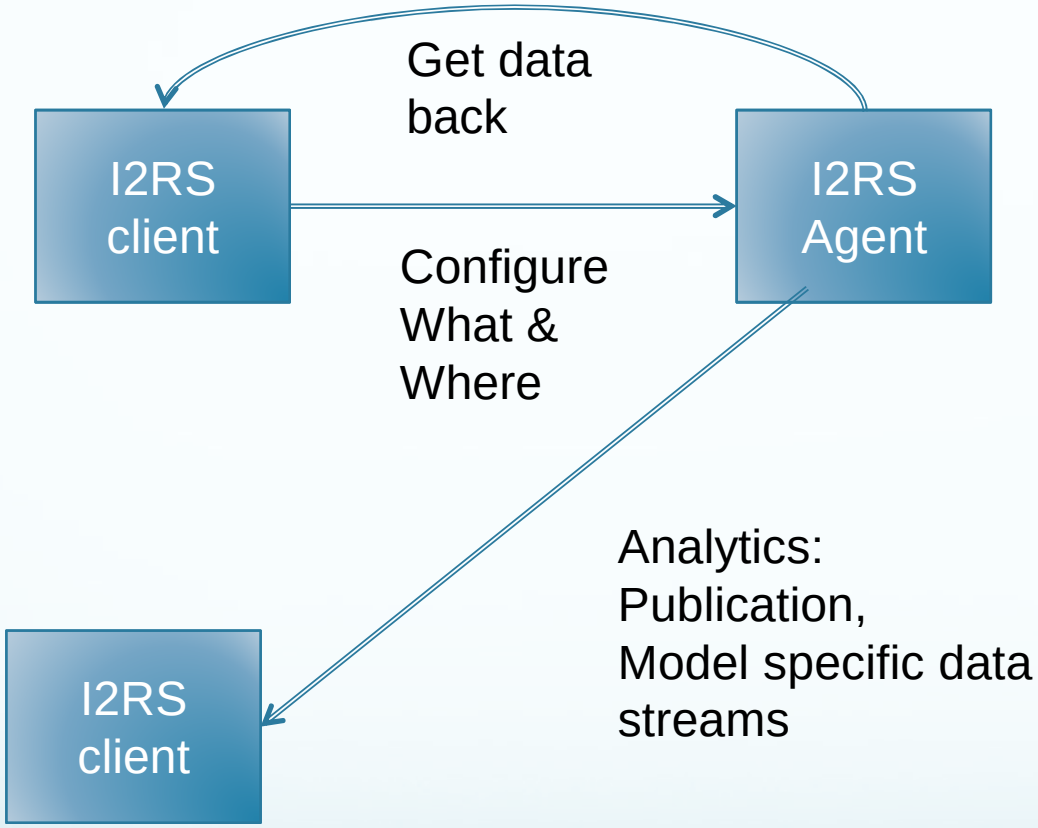
- Depend on Data Model + Use cases
- Method to get to one:
  - Propose to I2RS Data model + protocol
  - Security review
  - NETCONF/NETMOD review

Pub/Sub Requirements

Pub/Sub

- The I2RS interface should support user subscriptions to data with the following parameters: push of data synchronously or asynchronously via registered subscriptions…
  - Configuration of parameters via netconf
  - Distribution of data
- The I2RS interface (protocol and IMs) should allow a subscribe to select portions of the data model.
- Real time notification of events e.g. route installation and removal
  - 1-5 seconds

Pub/sub and other transports



Pub/Sub (cont.)

Requirements related to protocols

- The I2RS agent should be able to notify the client via publish or subscribe mechanism a BGP route change on a specific IP
  - Response: Notification/Subscription per Model and per item
- Can subscribe to the I2RS Agent's notification of critical node IGP events.
  - Response: Data model defines critical nature
- I2rs must be able to collect large data set from the network with high frequency and resolution with minimal impact to the device's CPU and memory
  - Response (from Alia): 2000/second

Pub/sub and tracing

- I2RS Agents should support publishing I2RS trace log information to that feed
- Draft-