# Ephemeral State

draft-ietf-i2rs-ephemeral-state-08.txt

## Interim & Email

- Last Requirement Document
- Authors Email  before WG LC

## Version -08 differences from -06

- Additions from -06.txt
  - Short description on re-use protocol (section 1)
  - Addition of Ephemeral-REQ-05 on resource constraints
- Changes
  - Removal of specific changes to Yang (section 3.4.1)
  - Change of NETCONF/RESTCONF sections to differentiate changes and required features
  - Changed PUB-SUB Requirements to 2 requirements specific to ephemeral

## I2RS as Re-use protocol

- I2RS WG selects features from YANG, NETCONF, and RESTCONF per version of the I2RS protocol (See sections 4, 5, and 6)
- I2RS WG proposes additions to YANG, NETCONF, and RESTCONF per version of the I2RS protocol for key functions (ephemeral state, protocol security, publication/subscription service, traceability),
- I2RS WG suggests protocol strawman as ideas for the NETCONF, RESTCONF, and YANG changes.

# Ephemeral-REQ-05

- Ephemeral state handling and notifications could increase need for CPU processing, data flow rates across a transport, or the rate of publication of data in a subscription or the logging for traceability.

- The I2RS Agent SHOULD have the ability to constraints for OAM functions operating to limit CPU processing, data rate across a transport, the rate of publication of data in a subscription, and logging rates;
- and the I2RS Agent SHOULD have the ability to prioritize some of the management data flows between the I2RS Agent and I2RS Client.
- In order to constrain resources needed, the I2RS Agent MAY also schedule data flows or split data flows unto multiple data flow streams.

# Changes to YANG

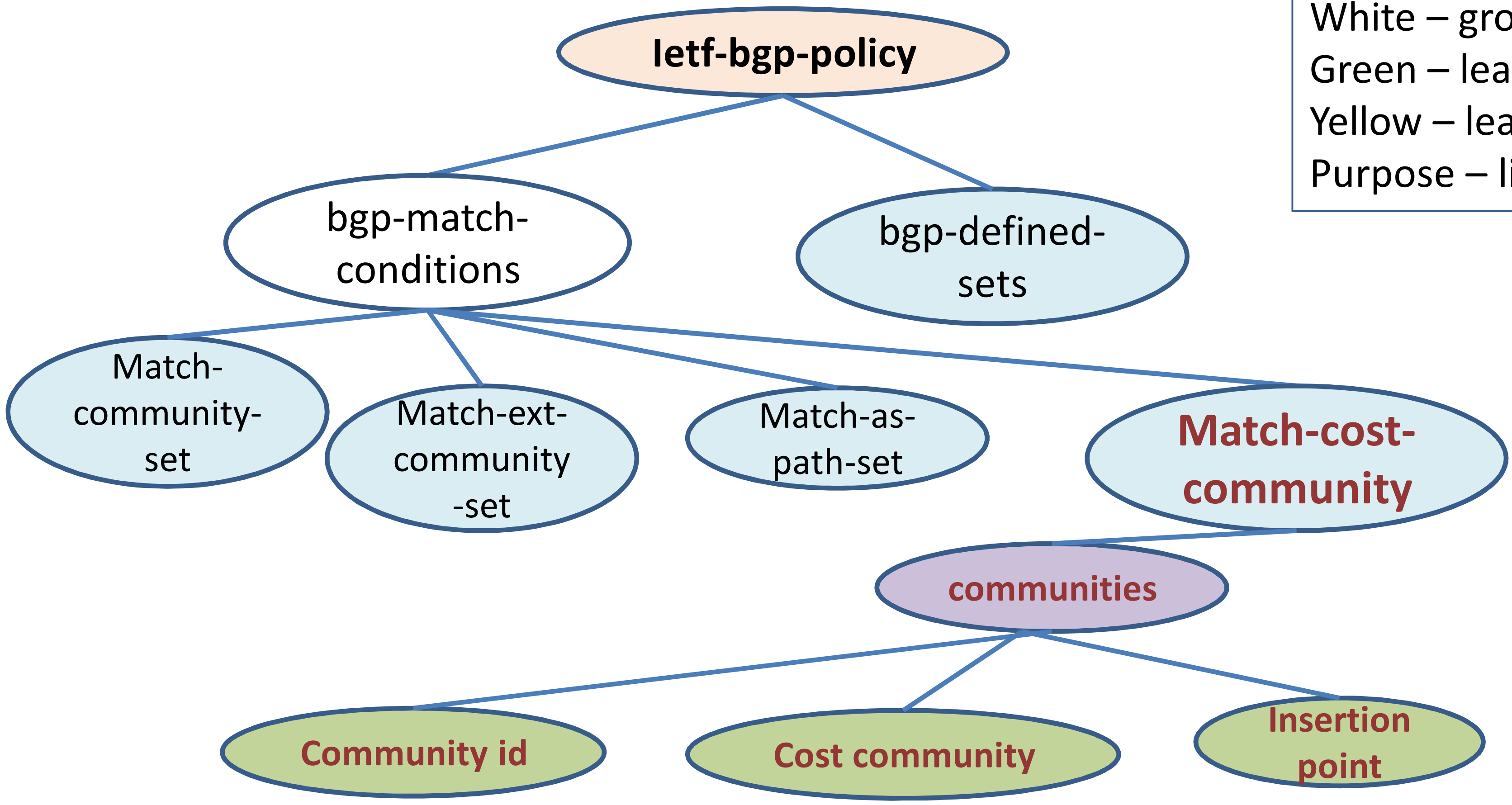
- Ephemeral-REQ-06: The ability to augment an object with appropriate YANG structures that have the property of being ephemeral. An object defined as:
  - Yang module (and the module's schema tree),
  - a schema node (**container,** leaf, **leaf-list**, choice, case, rpc, input, output, notifications, and anyxml),
  - submodule or components of a submodule (e.g. derived types, **groupings**, data node, RPCs, actions, and notifications).

# Stand-alone Ephemeral Modules and **Submodules**

- I2RS Yang Module
  - ietf-i2rs-rib
  - ietf-network
  - l3-unicast-igp-topology
  - ospf-topology
  - isis-topology
  - Ietf-l2-topology

- I2RS Proposed (WG Adoption)
- ietf-fb-rib
- ietf-fb-rib-types
- ietf-pkt-eca-policy

# Adding Ephemeral Container



**Legend**
Brown – module
Blue – container
White – grouping
Green – leaf
Yellow – leaf-list
Purpose – list

# Grouping:  bgp-global_config

```
module ietf-bgp-policy {
yang-version “1”;
namespace “urn:ietf:params:xml:ns:yang:ietf-bgp-policy”;
prefix bgp-pol:
import ietf-inet-types {prefix inet;}
import ietf-routing-policy {prefix rpol; }
Import ietf-policy-types { prefix pt;}
Import ietf-bgp-types { prefix bgp-types};
….
Grouping bgp-match-conditions {
   container match-community-set {  … }
   container match-ext-community-set { … }
   container match-as-path-set { … }
   container match-cost-community {
          ….
     }
```

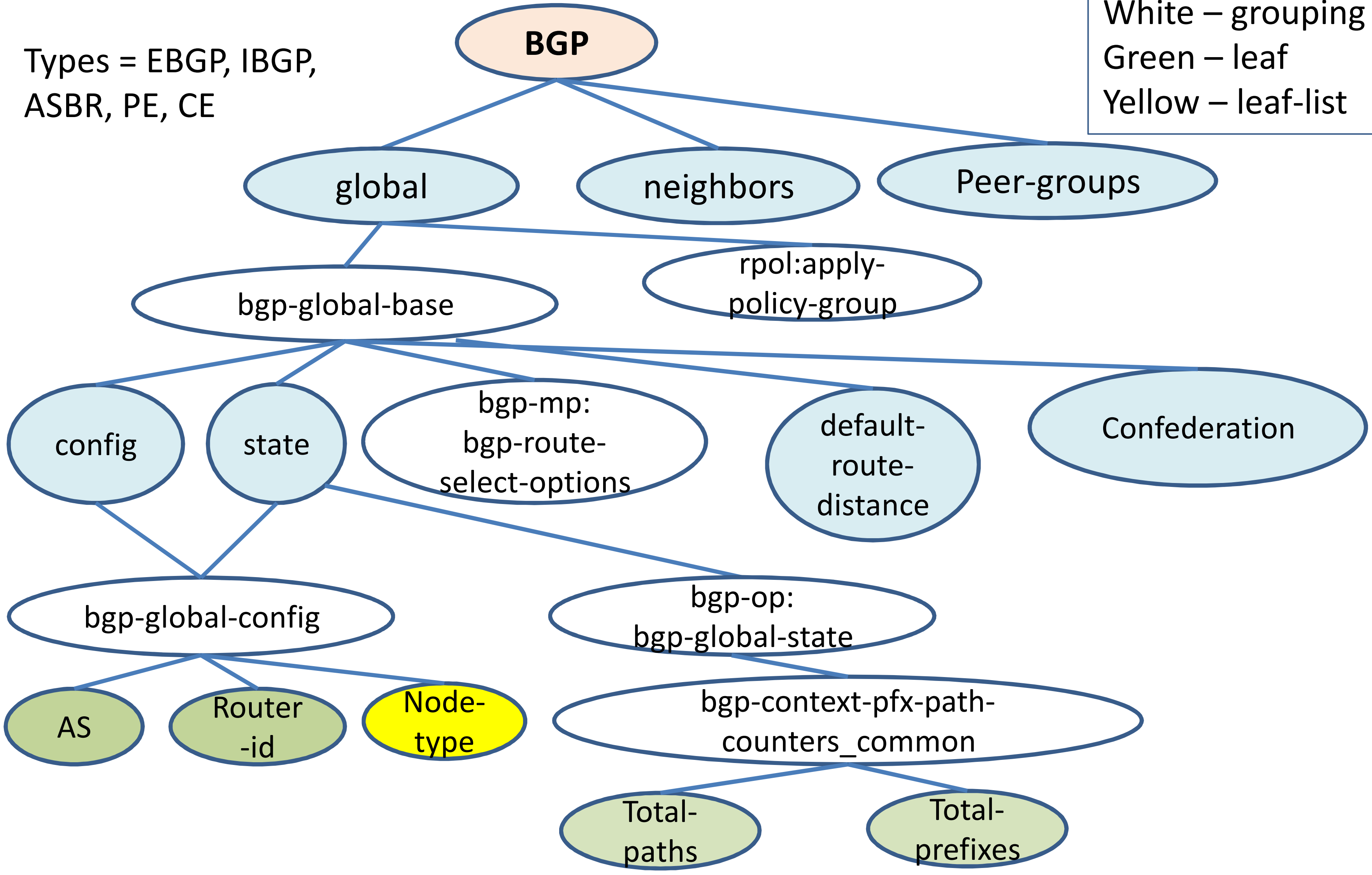
# Container Addition

```
container match-cost-community {
        if-feature i2rs-protocol;
        ephemeral true;
        list communities {
              key community-id;
             leaf community-id {
                  type uint8;
                  description “community id”;
              }
             leaf cost {
                  type uint32;
                   description “cost community”;
              }
             insertion-point  {
                 type uint8;
                  description “point of insertion”;
               }
        }
 }
```

# Adding Ephemeral to grouping

**Legend**
Brown – module
Blue – container
White – grouping
Green – leaf
Yellow – leaf-list

Types = EBGP, IBGP, ASBR, PE, CE



# Grouping: bgp-global_config

```
module ietf-bgp {
yang-version “1”;
namespace “urn:ietf:params:xml:ns:yang:ietf-bgp”;
prefix bgp:
…
grouping bgp-global_config {
Description “Global configuration options for the BGP router”;
  leaf as {
        type inet:as-number;
        mandatory “true:
        description
    “local autonous system number of the router.
      Uses the 32-bit as-number type from model in RFC6991.:;
   }
 leaf router-id {
     type inet:ipv4-address;
     description
      “Router id of the router, expressed as an
        32-bit value, IPv4 address.”;
   }
}
```

# Ephemeral Group Addition

```
model ietf-i2rs-bgp-config {
yang-version “1.1”;
Namespace
“urn:ietf:params:xml:yang:ietf-i2rs-bgp-
config;”
prefix “i2rs-bgp-cfg”;
import ietf-bgp {prefix bgp;}
Identity i2rs-bgp-node-type {
description “type of node (IBGP,
EBGP(ASBR, PE, CE)”;
};
identity NT-IBGP {
 base i2rs-bgp-node-type;
 description “Node with IBGP peer”;
}
```

```
identity NT-ASBR  {
 base i2rs-bgp-node-type;
 description “node with EBGP as ASBR”;
}
identity NT-PE {
 base i2rs-bgp-node-type
 description “node with EBGP as PE”;
}

Identity NT-CE {
  base i2rs-bgp-node-type
  description “node with EBGP as CE”;
}
typedef bgp-i2rs-node-types {
 type identityref {
   base i2rs-bgp-node-type;
  }
description “I2RS node types”;
}
```

## Add leaf-list

```
feature i2rs-protocol {
         description "I2RS protocol supported";
}

augment "/bgp/global/bgp-global-base/config/bgp-global_config
       leaf-list  node-type {
        if-feature I2rs-protocol;
        ephemeral true;
        type bgp-i2rs-node-types;
        description "Node type on node";
    }
}
```

Yang 1.1 reference page 7.15 – pages 93-96

# Ephemeral-REQ-07: NETCONF changes

1. protocol version support for I2RS modifications - (e.g. I2RS version 1)
2. support for ephemeral model scope indication - ephemeral only, mixed-config-ephemeral, mixed-opstate-ephemeral
3. I2RS multiple message support - supports only I2RS "all or nothing“ aka NETCONF "roll-back-on-error".
4. support for the following transports protocol supported: "TCP", "SSH", "TLS", **and non-secure transport**
5. ability to restrict non-secure transport support to specific portions of a data models marked as valid to transfer via insecure protocol.

# Ephemeral-REQ-07: NETCONF changes (6-7)

6. ephemeral state overwriting of configuration state MUST be controlled by the following policy knobs
   – ephemeral configuration overwrites local configuration (true/ false; normal value: true),
   – Update of local configuration value supercedes and overwrites the ephemeral configuration (true/false; normal value: false).

7. The ephemeral overwriting to local configuration described in (6) above is considered to be the composite of all ephemeral values by all clients. Some may consider this approach as a single pane of glass for ephemeral state.

# Knobs

| Knob 1 | Normal | Option 2 | Option 3 | Option 4 |
|---|---|---|---|---|
| Knob 1: Ephemeral configuration overwrites local configuration | **True** | False | False | True |
| Knob 2: Update of local configuration value supersedes and overwrites ephemeral configuration | **False** | True | False | True |
| **Results** | **Ephemeral stays until removed** | Ephemeral never gets added if local config exists | | Ephemeral gets added, if local config changes is lost |
| **Why** | **Normal** | Turn off ephemeral overwrite of local-config | | Desire config updates to win |

# Ephemeral-REQ-07: NETCONF Changes (8-9)

8. The ephemeral state must support notification of write conflicts using the priority requirements defined in section 3.7 below in requirements Ephemeral-REQ-09 through Ephemeral-REQ-14).
9. Ephemeral data stores SHOULD not require support interactions with writable-running, candidate data store, confirmed commit, and a distinct start-up capability,

# Ephemeral-REQ-08: RESTCONF changes

1. protocol version support for I2RS modifications - (e.g. I2RS version 1)
2. support for ephemeral scope  -  ephemeral only,  mixed config (ephemeral config + config), mixed opstate (ephemeral opstate + opstate).
3. support for the following transports protocol supported:
   – “http over TLS”, and
   – “http used in non-secure fashion”.
   – RESTCONF should be able to expand the supported transports.
4. Ability of RESTCONF to support an insecure transport for specific portions of data models.
5. Support for development of RESTCONF based yang pub/sub

## Ephemeral-REQ-07: RESTCONF changes (7-8)

6. ephemeral state overwriting of configuration state MUST be controlled by the following policy knobs
   – ephemeral configuration overwrites local configuration (true/ false; normal value: true),
   – Update of local configuration value supersedes and overwrites the ephemeral configuration (true/false; normal value: false).

7. The ephemeral overwriting to local configuration described in (6) above is considered to be the composite of all ephemeral values by all clients. Some may consider this approach as a single pane of glass for ephemeral state.

## Ephemeral-REQ-07: RESTCONF Changes (8-9)

8. The ephemeral state must support notification of write conflicts using the priority requirements defined in section 7 below in requirements Ephemeral-REQ-10 through Ephemeral-REQ-13).
9. Ephemeral data stores SHOULD not require support interactions with writable-running, candidate data store, confirmed commit, and a distinct start-up capability,

## Use NETCONF/RESTCONF features

- Encodings: XML or JSON
- Transports: SSH, TLS, TCP (non-secure)
- NETCONF features:
  - Yang pub-sub
  - Yang module library
  - Call-home
  - Zero touch
  - Server-module

- Encodings: XML or JSON
- Transports: http over TLS, http (non-secure)
- RESTCONF features:
  - Yang module library
  - Call-home
  - Zero touch
  - Server-module

## PUB-SUB Ephemeral Requirements

- Pub-Sub-REQ-01: The Subscription Service MUST support subscriptions against ephemeral data in operational data stores, configuration data stores or both.

- Pub-Sub-REQ-02: The Subscription Service MUST support filtering so that subscribed updates under a target node might publish only ephemeral data in operational data or configuration data, or publish both ephemeral and operational data.

## BACKUP

## Multi-headed Control

10. I2RS data nodes MAY I2RS client identity and not the effective priority at the time the data node is stored. Priority MAY be dynamic changed by AAA – as along as collisions are handled per 10, 11, 12
11. Collision is considered an Error. Notification MUST Be sent to the original client to deal with issues surrounding collision.
12. Multi-headed control is not tied to ephemeral state
13. Two clients with same priority, the first client [MUST] Wins.

## Multi-headed Control

14. No multi-message commands SHOLD cause errors to be inserted into the I2RS ephemeral data-store