An Overview of gNMI Support in Stratum

Yi Tseng
ONF
Outline

- gNMI, OpenConfig model, and Stratum model
- Stratum Config Monitoring Service
  - Stratum Chassis Config
  - gNMI publisher
  - Tree structure for config and state
  - The Root Path
- Yang model to openconfig.proto
- Demo
- Contribute to the Stratum gNMI service
gNMI (gRPC Network Management Interface)

Generic API to read and write configuration state
Suitable for any tree-based data model
• YANG as a possible data model

module openconfig-interfaces {
    ...
    container interfaces {
        ...
        list interface {
            key "name";
            ...
            container config {
                ...
            }
        }
    }
}

service gNMI {
    rpc Capabilities
    rpc Get
    rpc Set
    rpc Subscribe
}

gNMI path
/interfaces/interface[name=eth0]/config/.....
gNMI Requests

Get /interfaces/interface[?name=eth0]/state/ifindex
OpenConfig model

Vendor-Neutral Data Models for Configuration and Management that are supported natively by network hardware and software devices.

Represents a variety of network operators’ use cases

Only a subset are relevant to Stratum (e.g., interfaces, system, …)

Stratum also uses some augmentations defined in openconfig/hercules

Vendors can also provide augmentations and deviations on top of this.

Updated: 05-15-2018
Stratum model

OpenConfig
- Interfaces
- Platform
- Lcap
- Vlan
- System
- QoS

Stratum
- Interfaces
- Platform
- Lcap
- Vlan
- System
- QoS
- Vendor specific

Deviations/Augments

https://github.com/openconfig/hercules/tree/master/yang
Stratum architecture recap
Stratum Config Monitoring Service
Stratum Config Monitoring Service

Implementation of gNMI service (Get, Set, …)
Pushes the **Stratum Chassis Config** to Switch Broker Interface and Gnmi Publisher
Pass gNMI requests to the gNMI Publisher.
Manage gRPC streams for gNMI subscriptions.
A data structure that encapsulates the config pushed to the entire chassis.

“Chassis” refers to the a switching box with one or more switching nodes.
gNMI Publisher

Creates and stores the Tree which includes **config** and **states**.

Init/Update/Replace/Delete tree node(s).

Subscribe tree node(s).

Process events from Switch Broker Interface.
Create The Yang Parse Tree

The gNMI publisher creates the tree data structure based on the Chassis Config provided by user.
It initialize all necessary tree nodes with tree node handlers.
Config/State stored in Stratum

/interfaces/interface[name=eth0]/state/ifindex
gNMI service implementation calls handler to retrieve the value from node or update the tree node.
The Root (/) path

The root path is a special path which is for set and get of the entire chassis config.

The Chassis Config needs to be converted to the OpenConfig protobuf message (openconfig.proto) before set or get.

Stratum uses Yang to Protobuf compiler to compile the OpenConfig Yang model to protobuf format.
module openconfig-interfaces {
  ...
  container interfaces {
    ...
    list interface {
      key "name";
      ...
      leaf oper-status {
        ...
      }
    }
  }
}

message Device {
  ...
  message InterfaceKey {
    string name = 1;
    Interface interface = 2;
  }
  repeated InterfaceKey interface = ....;
}

message Interface {
  ...
  OperStatus oper_status = .....;
  ...
}
Demo

Stratum Dummy Switch

gNMI CLI

Dummy Switch Box API
Contribute to Stratum gNMI service

Add new paths from the OpenConfig model

Add new test cases for TestVector
  Test Vector Framework for Stratum Enabled Switches - Abhilash Endurthi, You Wang (9/12, 3:00 pm)

Add new platform component support
  Stratum’s Phal Attribute DB: What Is It Good For? - Craig Stevens (3:00 pm, Next session)
  Stratum’s New Capabilities: Optical Transport Support for Cassini Chassis - Leonid Khedyk (5:00 pm)
Thank you!