STRATUM

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

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

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

OpenConfig model

gNMI path

```
/interfaces/interface[name=eth0]/config/…..
```

gNMI Requests

```
path {
 elem { name: "interfaces"  }
 elem {
          name: "interface"
          key { key: "name" value: "eth0" }
 }
 elem { name: "state" }
 elem { name: "ifindex" }
}
Type: STATE
encoding: PROTO
```

```
notification {
 timestamp: 1568184227017361000
 update {
  path {
   elem { name: "interfaces" }
   elem {
    name: "interface"
    key { key: "name" value: "eth0" }
   }
   elem { name: "state" }
   elem { name: "ifindex" }
  }
  val { uint_val: 1 }
 }
}
```

Get /interfaces/interface[name=eth0]/state/ifindex

OpenConfig model

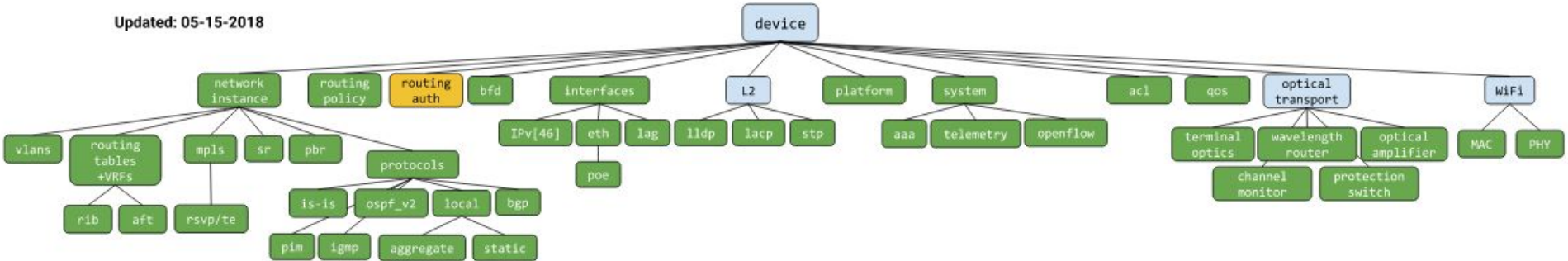
Vendor-Neutral Data Models for Configuration and Management that are supported natively by network hardware and software devices.

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.



Stratum model

OpenConfig

- Interfaces
- Platform
- Lacp
- Vlan
- System
- QoS

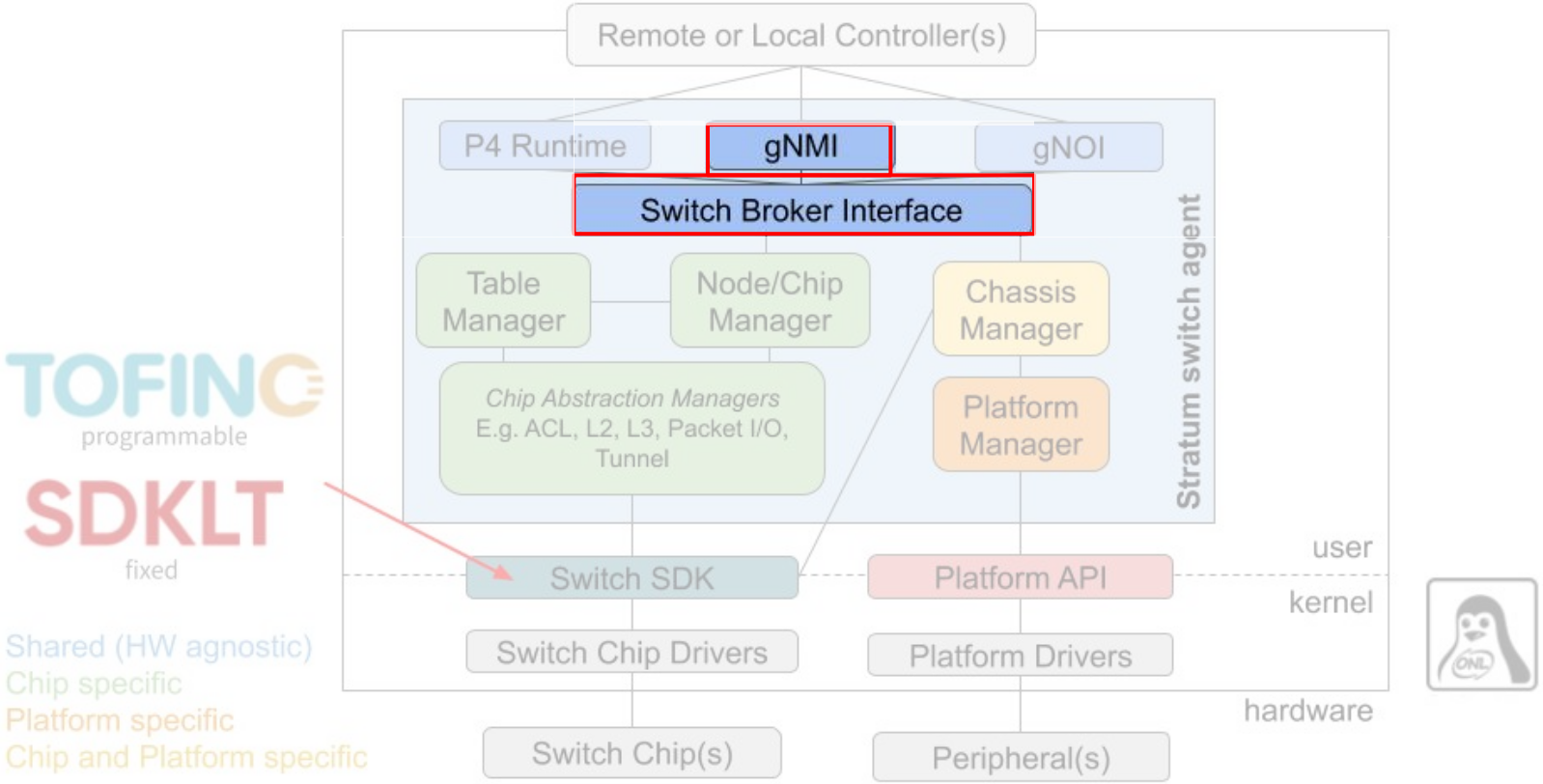
Deviation/Augment

Stratum

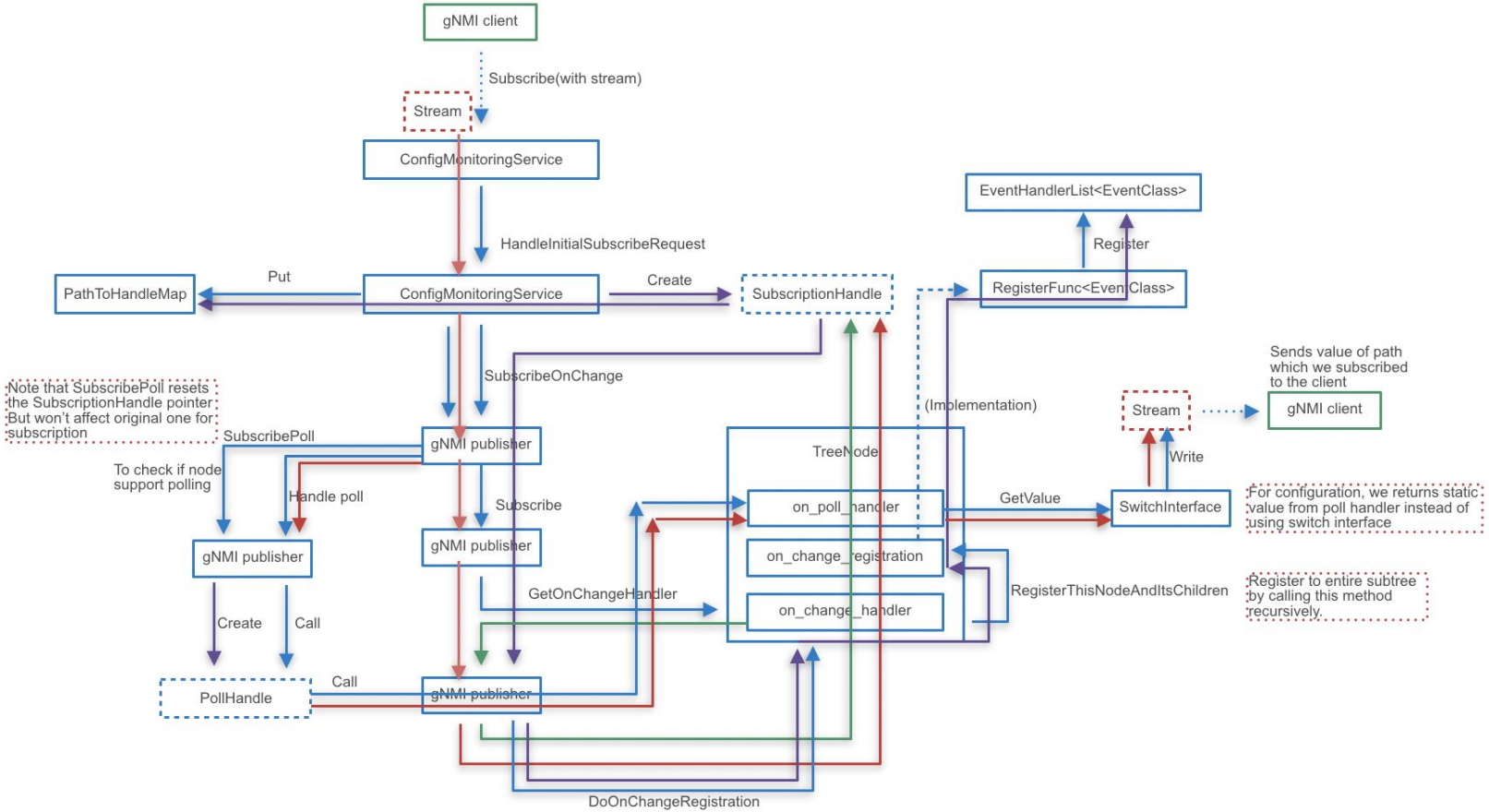
- Interfaces
- Platform
- Lcap
- Vlan
- System
- QoS
- Vendor specific

https://github.com/openconfig/hercules/tree/master/yang

Stratum architecture recap



Stratum Config Monitoring Service


gNMI client
Subscribe(with stream)
Stream
ConfigMonitoringService
HandleInitialSubscribeRequest
Put
PathToHandleMap
ConfigMonitoringService
Create
SubscriptionHandle
EventHandlerList<EventClass>
Register
RegisterFunc<EventClass>
Sends value of path which we subscribed to the client
Note that SubscribePoll resets the SubscriptionHandle pointer But won't affect original one for subscription
SubscribeOnChange
(Implementation)
Stream
gNMI client
SubscribePoll
gNMI publisher
TreeNode
Write
To check if node support polling
Handle poll
Subscribe
on_poll_handler
GetValue
SwitchInterface
For configuration, we returns static value from poll handler instead of using switch interface
gNMI publisher
gNMI publisher
on_change_registration
GetOnChangeHandler
RegisterThisNodeAndItsChildren
on_change_handler
Register to entire subtree by calling this method recursively.
Create
Call
Call
PollHandle
gNMI publisher
DoOnChangeRegistration

Stratum Config Monitoring Service

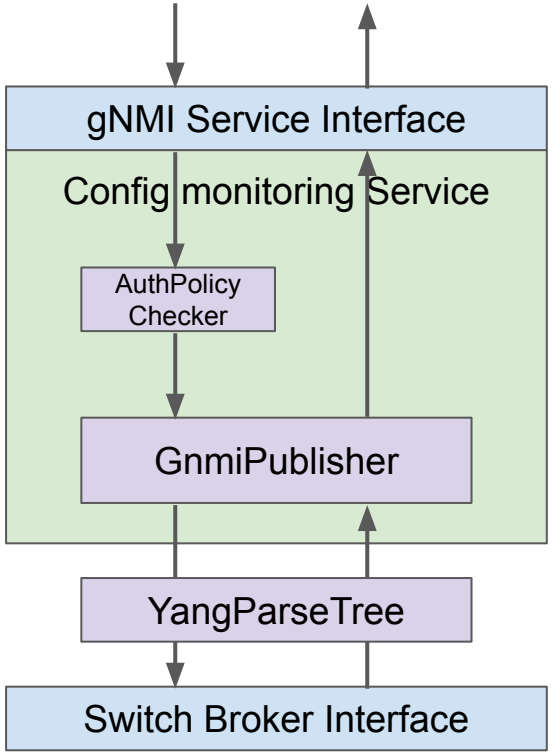
Implementation of gNMI service (Get, Set, …)

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.



Stratum Chassis Config

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.

```
chassis {
  platform: PLT_P4_SOFT_SWITCH
  name: "dummy switch 1"
  config_params: {
  }
}
nodes {
  id: 1
  name: "node 1"
  slot: 1
  index: 1
  flow_params {
  }
}
singleton_ports {
  id: 1
  name: "1/1/1"
  slot: 1
  port: 1
  channel: 1
  speed_bps: 100000000000
  node: 1
}
```

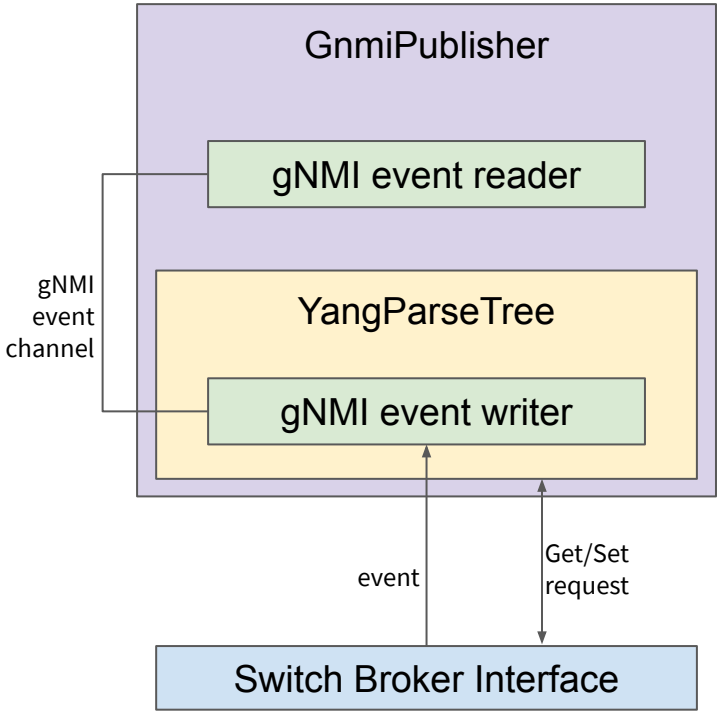
gNMI Publisher

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

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

Subscribe tree node(s).

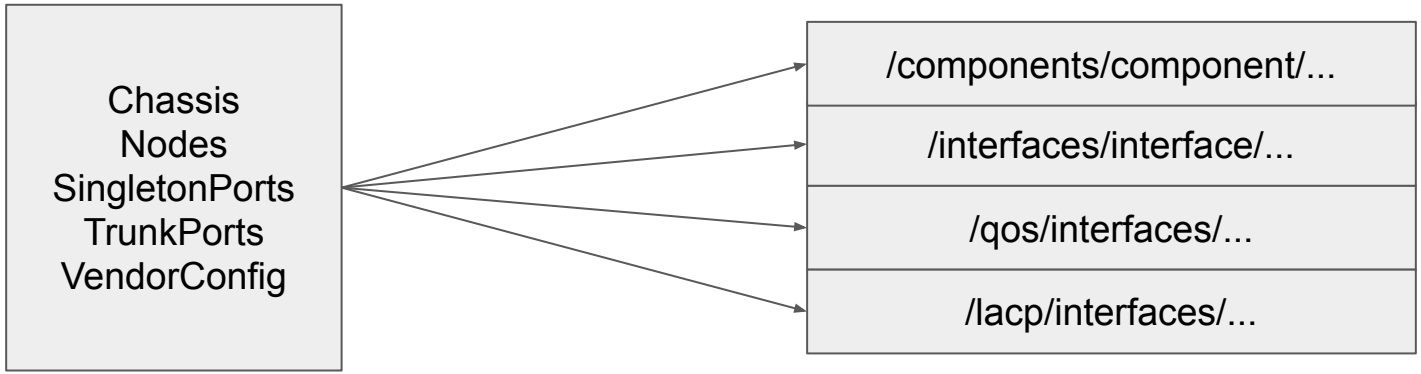
Process events from Switch Broker Interface.



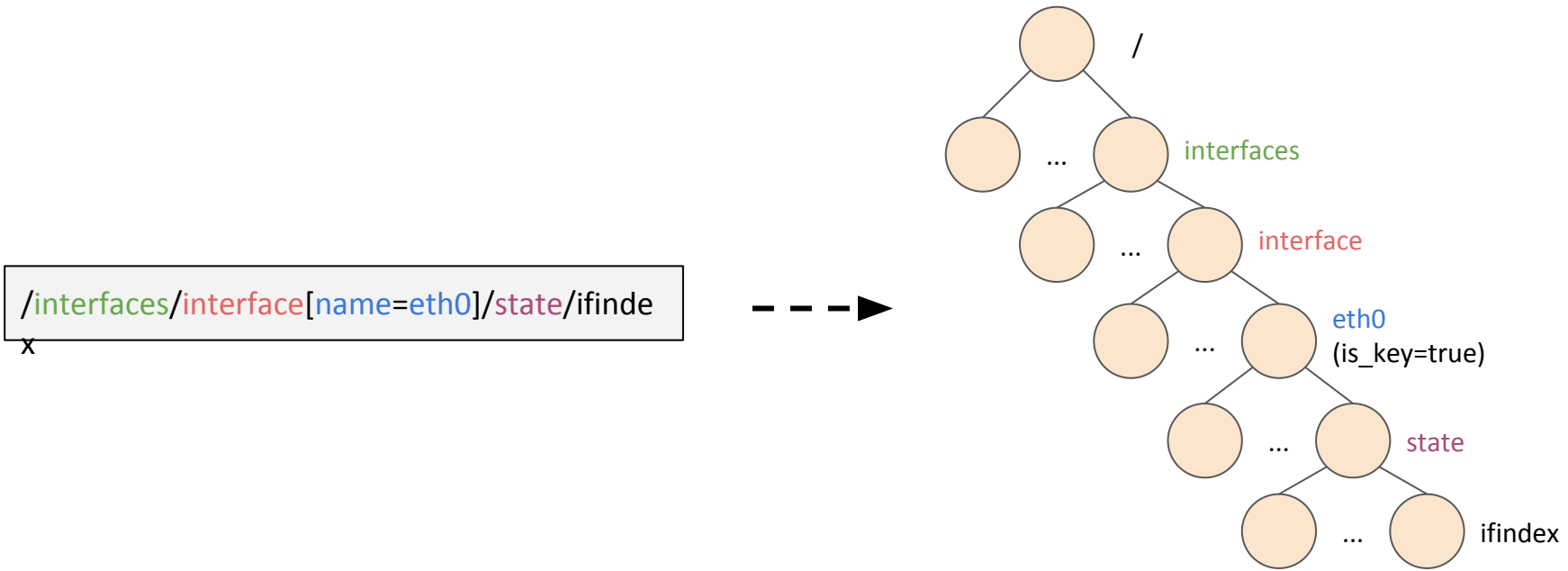
Create The Yang Parse Tree

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.

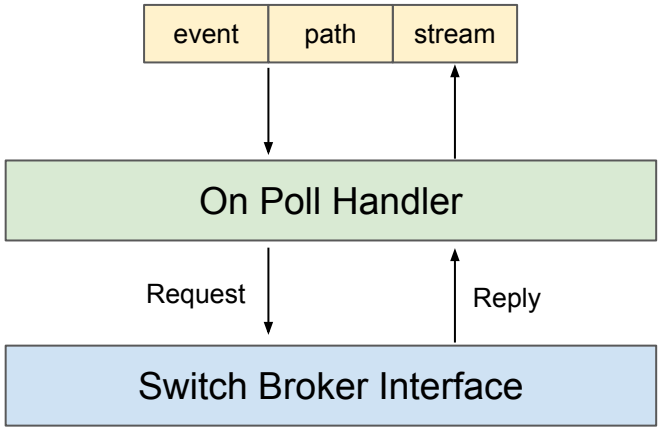


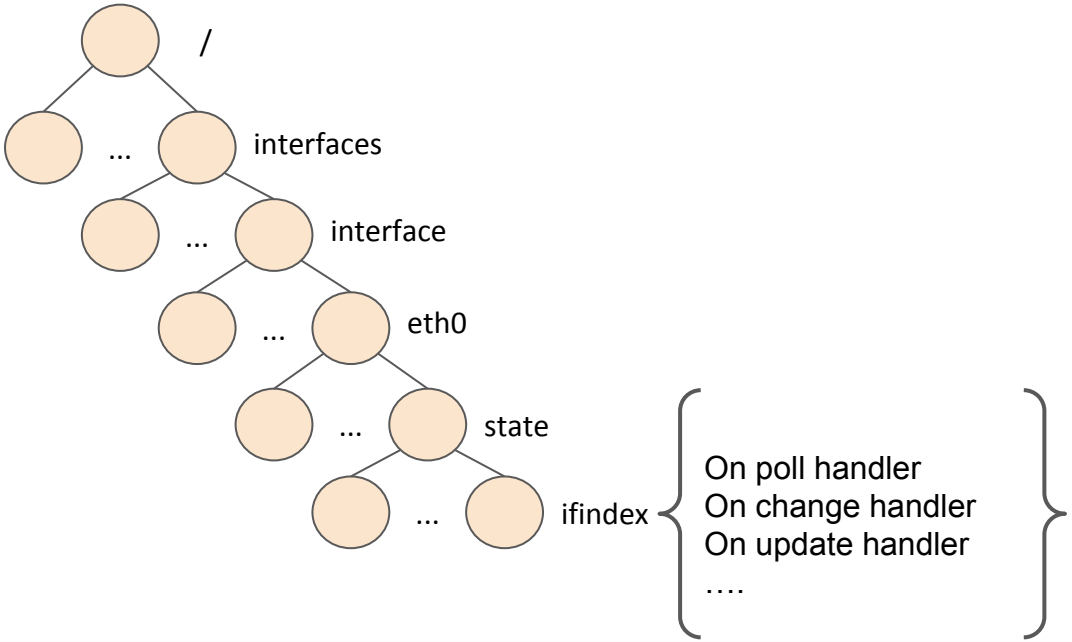
Config/State stored in Stratum



Config/State stored in Stratum

gNMI service implementation calls handler to retrieve the value from node or update the tree node.





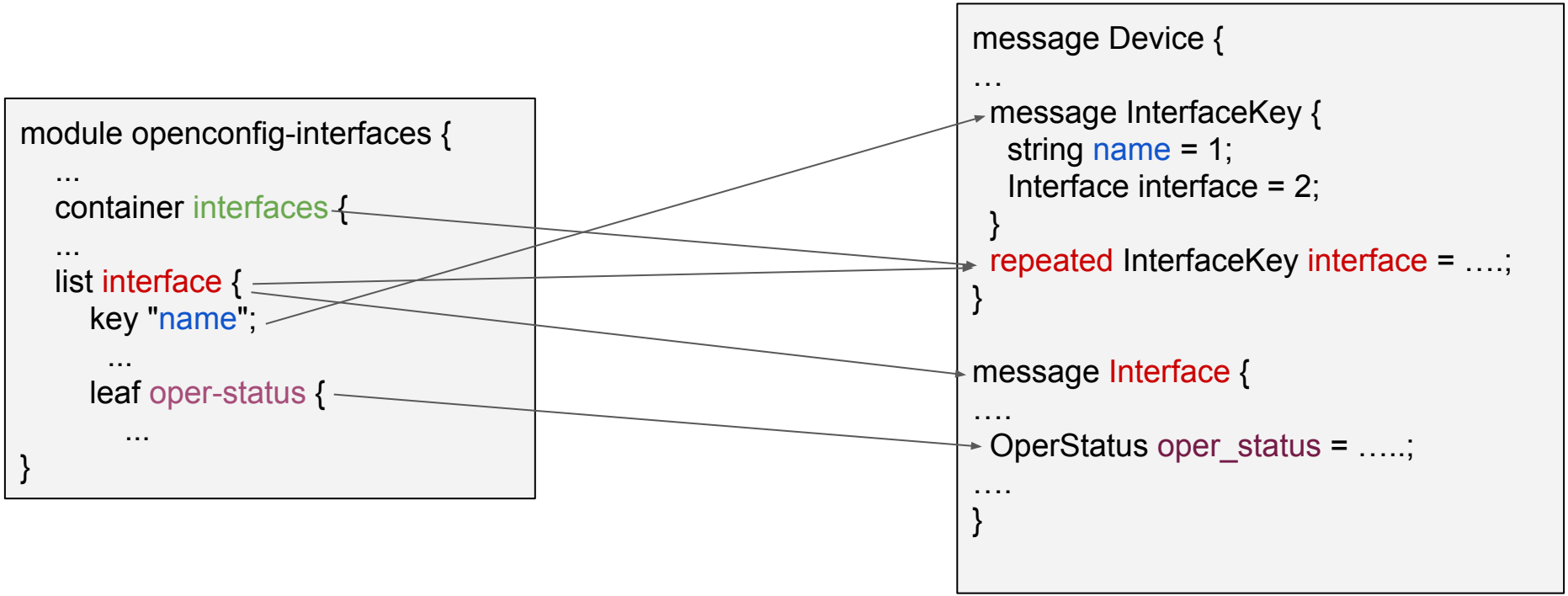
The Root (/) path

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.

Generate Protobuf from Yang Models

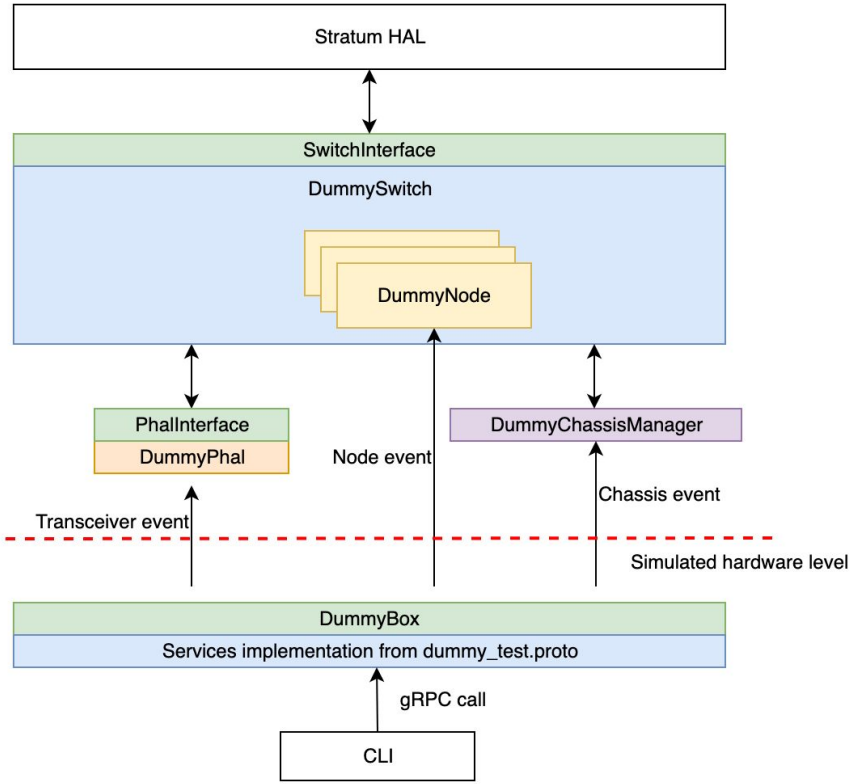


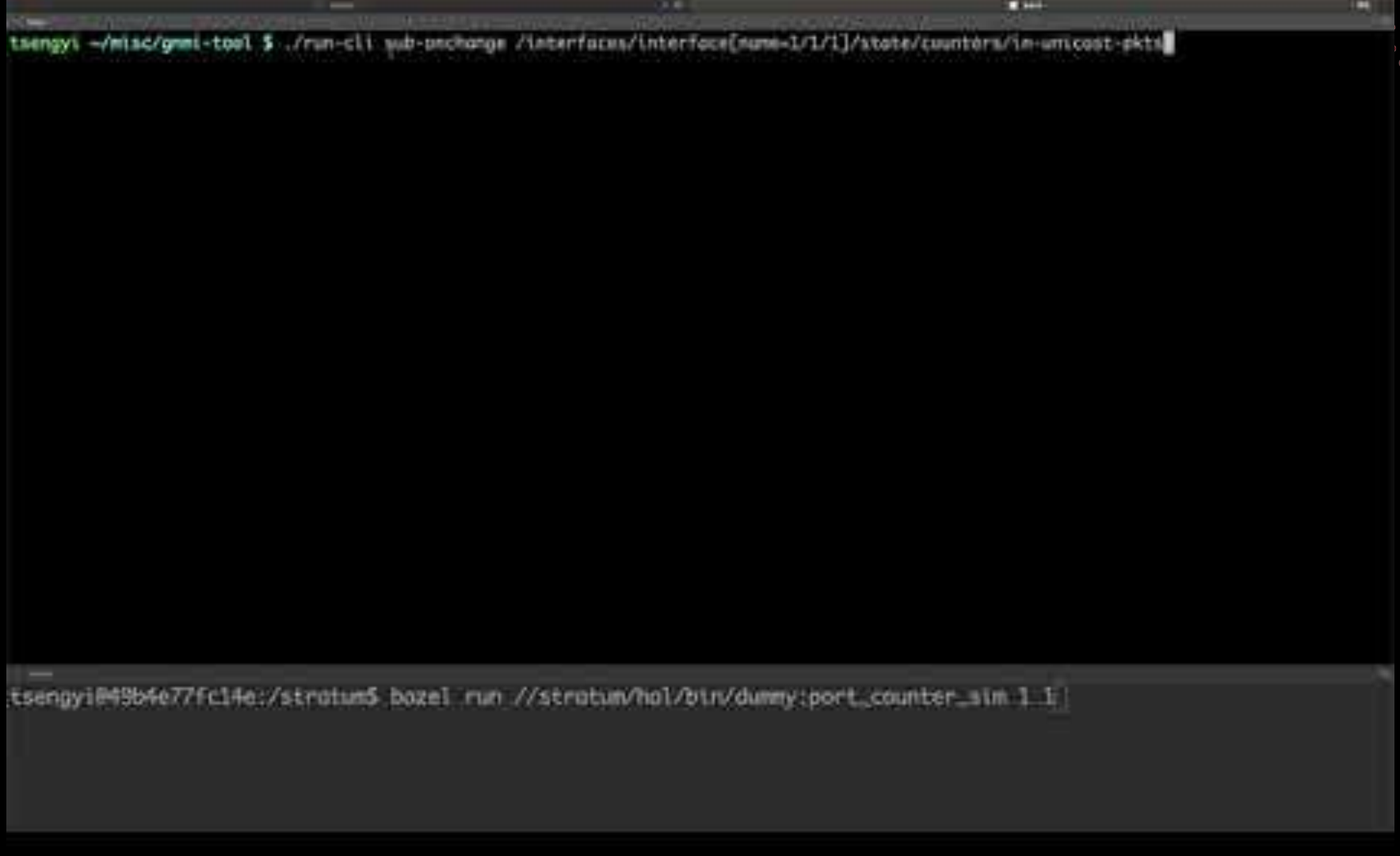
Demo

Stratum Dummy Switch

gNMI CLI

Dummy Switch Box API




tsengyi ~/misc/gnmi-tool $ ./run-cli sub-onchange /interfaces/interface[name=1/1/1]/state/counters/in-unicast-pkts
tsengyi@49b4e77fc14e:/stratum$ bazel run //stratum/hal/bin/dummy:port_counter_sim 1 1

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!