Development Process and Testing

Session 4: 3:30pm - 5pm
In this session

- What does a typical SEBA development loop look like?
- How do we test functionality end-to-end? How do we test system scale?
- What are the next steps for SEBA? (last 30 minutes)
Development Loop

- Make changes to component source
- Pass unit tests for that component
- Build new container image for the component
- Deploy new container in SiaB
- Pass integration/E2E tests
- Lab #4 will walk through this process
Cord-Tester

- The cord-tester is a test-automation framework written in Robot Framework and Python
- Test suites to validate functionality, regression, and stability of various components of CORD (xos, onos, voltha, siab, ponsim, etc.)
- Shared libraries between multiple test suites (physical pod, siab, xos-api-tests, integration-tests, etc)
- Current E2E test suites based on the ATT-Workflow
Example SEBA Test w/Framework Reference

- Push required configurations (tosca or json)
  - Tosca files (SEBA for various components - fabric, olt, subscribers etc)
  - Json (inputs for POD configurations)
- Validate states in voltha, onos and NEM for devices and subscribers
  - Checks in voltha and onos cli by sending various commands
- Authenticate
  - Login to the subscriber/specified host and send auth request via wpa_supplicant
  - Validate command output status
  - Validate NEM states
- DHCP
  - Send dhclient commands from the specified host
  - Validating command success states, verifying assigned address on the interface
  - Validate NEM states
- Pings
  - Send and validate ping requests
Current SEBA Tests
Test Case #1

Full Name: Validate ONU in Correct Location

Documentation: Validates E2E Ping Connectivity and object states for the given scenario. Configure whitelist with correct ONU location. Validate successful authentication/DHCP/E2E ping

Tags: latest, multicast, stable, test

Start/End/Elapsed: 20190904 08:10:04.185 / 20190904 08:13:50.067 / 00:03:45.882

Status: PASS (crITICAL)

**Keyword** Subscriber Ready to Authenticate

Start/End/Elapsed: 20190904 08:10:04.187 / 20190904 08:10:04.465 / 00:00:00.268

- **Keyword** Wait Until Keyword Succeeds 60s, 15s. Validate ONU Statuses, ACTIVE, ENABLED, $(onu_device)
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate ATT Workflow Driver, SL, ENABLED, AWAITING, $(onu_device), ONU has been validated - Awaiting Authentication
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Subscriber Status, awaiting-auth, $(onu_device)
- **Subkeyword** Validate Authentication True, eth0, ups_supplicant.conf, $(kube_node_ip), $(local_user), $(local_pass), K8S, $(RG_CONTAINER)

**Keyword** Subscriber Service Chain Created

Start/End/Elapsed: 20190904 08:10:10.145 / 20190904 08:10:22.059 / 00:00:11.913

- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate ATT Workflow Driver, SL, ENABLED, APPROVED, $(onu_device), ONU has been validated - Authentication succeeded
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Subscriber Status, enabled, $(onu_device)
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Subscriber Service Chain, $(onu_device), True
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Fabric CrossConnect SI, $(e_tag), True
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate XConnect in ONOS, True
- **Subkeyword** Validate DHCP and Ping True, True, eth0, $(e_tag), $(c_tag), $(dist_host_ip), $(kube_node_ip), $(local_user), $(local_pass), K8S, $(RG_CONTAINER)
- $(subscriber_id) = Subscriber Retrieve Subscriber $(c_tag)
- **Link** CORD Put $(VOLT_SUBSCRIBER), ("status":disabled), $(subscriber_id)

**Keyword** No Subscriber Service Chain

Start/End/Elapsed: 20190904 08:10:46.167 / 20190904 08:10:53.382 / 00:00:07.215

- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Subscriber Service Chain, $(onu_device), False
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Fabric CrossConnect SI, $(e_tag), False
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate XConnect in ONOS, False
- **Subkeyword** Validate DHCP and Ping False, False, eth0, $(e_tag), $(c_tag), $(dist_host_ip), $(kube_node_ip), $(local_user), $(local_pass), K8S, $(RG_CONTAINER)
- **Link** Restart RG Pod
- **Link** CORD Put $(VOLT_SUBSCRIBER), ("status":waiting-auth), $(subscriber_id)
- **Subkeyword** Validate Authentication True, eth0, ups_supplicant.conf, $(kube_node_ip), $(local_user), $(local_pass), K8S, $(RG_CONTAINER)

**Keyword** Subscriber Service Chain Created

Start/End/Elapsed: 20190904 08:12:15.116 / 20190904 08:12:25.060 / 00:00:11.944

- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate ATT Workflow Driver, SL, ENABLED, APPROVED, $(onu_device), ONU has been validated - Authentication succeeded
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Subscriber Status, enabled, $(onu_device)
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Subscriber Service Chain, $(onu_device), True
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate Fabric CrossConnect SI, $(e_tag), True
- **Keyword** Wait Until Keyword Succeeds 60s, 2s. Validate XConnect in ONOS, True
- **Subkeyword** Validate DHCP and Ping True, True, eth0, $(e_tag), $(c_tag), $(dist_host_ip), $(kube_node_ip), $(local_user), $(local_pass), K8S, $(RG_CONTAINER)

**Keyword** Test Cleanup
Test Clean-up

- **TEARDOWN**: Test Cleanup
- **Documentation**: Restore back to initial state per each test
- **Start / End / Elapsed**: 20190910 07:10:25.543 / 20190910 07:11:25.421 / 00:00:59.778

  - **keyword**: GET VOLTHA Status
  - **keyword**: GET ONOS Status
  - **keyword**: Kubernetes Log Kubernetes Containers Logs Since Time $(datetime), $(container_list)
  - **keyword**: Wait Until Keyword Succeeds 60s, 2s, Clean Up Objects, $(ATT_WHITELIST)
  - **keyword**: Wait Until Keyword Succeeds 30s, 2s, Validate ONU States, UNKNOWN, DISABLED, $(onu_device)
  - **keyword**: Wait Until Keyword Succeeds 30s, 2s, Validate ATT Workflow Driver SI, DISABLED, AWAITING, $(onu_device)
  - **keyword**: Wait Until Keyword Succeeds 60s, 2s, Clean Up Objects, $(VOLT_SUBSCRIBER)
  - **keyword**: Wait Until Keyword Succeeds 60s, 2s, Validate Subscriber Service Chain, $(onu_device), False
  - **keyword**: Wait Until Keyword Succeeds 60s, 2s, Validate Fabric CrossConnect SI, $(s_tag), False
  - **keyword**: Restart RG Pod
SEBA Roadmap
Roadmap

SEBA 1.0 (Jan 7th 2019)
- HSIA service
- AT&T workflow
- VOLTHA 1.6
- New CORD platform (6.1)

SEBA 2.0 alpha (July 3rd 2019)
- Technology & Speed Profile
- AT&T workflow, HSIA service
- VOLTHA 1.7, BAL 2.6

SEBA 2.1 (Sept/Oct 2019)
- BAL 3.1
- ONOS & VOLTHA FCAPS

SEBA 3.0 & Beyond
- VOLTHA 2.x
- Embedded BNG
- Deutsche Telekom/Turk Telecom Workflows

SEBA 3.0 & Beyond
- VOLTHA 2.x
- Embedded BNG
- Deutsche Telekom/Turk Telecom Workflows
SEBA Modularity

ONAP/OSAM

Abstract OLT

NEM Orchestration (XOS)

NEM Monitoring (Prom, Grafana, ELK)

Access apps

Fabric apps

ONOS

VOLTHA

Trellis

Kafka

K8s

OS

ONU

OLT

AGG switch

Compute Nodes
# Accelerating SEBA: MVP Requirements & Gaps

https://docs.google.com/document/d/1eM-Sdy4dAQVm9oP1i7V7H_LXKbyD2S7julZOdFiPmXk/edit?ts=5d5dad90#

<table>
<thead>
<tr>
<th>Features desired by</th>
<th>YE2019</th>
<th>YE2019</th>
<th>YE2019</th>
</tr>
</thead>
<tbody>
<tr>
<td>Production with commercial subscribers</td>
<td>End 2019</td>
<td>End Q1 2020</td>
<td>End Q2 2020</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Features (docs, tests &amp; API) (must include openolt)</th>
<th>TT</th>
<th>DT</th>
<th>ATT</th>
<th>Gap (Y=Yes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSIA</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>N</td>
</tr>
<tr>
<td>EAPOL</td>
<td></td>
<td></td>
<td>✓</td>
<td>N</td>
</tr>
<tr>
<td>Subscriber DHCP</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>multi-gem</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Coming to voltha 2.x</td>
</tr>
<tr>
<td>VoIP, VoD</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>N</td>
</tr>
<tr>
<td>GPON (BAL3.1 dep)</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>in the works (M)</td>
</tr>
</tbody>
</table>
VOLTHA 2.x Integration

- VOLTHA 2.x was not at feature parity when SEBA 2.0 alpha was released.
- Community working extensively on stabilization of VOLTHA codebase, brigade focused on this.
- Current plan is to integrate when feature parity and stability goals are reached.
Key VOLTHA 2.x Changes

- Significant portions rewritten in go
- New High Availability model
  - disaggregated core: api-server, rwcores, rocores
  - active-active to ensure fast failover
- New event and performance metric format
  - Protobuf-encoded Kafka messages
VOLTHA 2.x: architecture overview

- OF Agent
- CLI
- Affinity Router (gRPC Server / Proxy)
- RO Core
- RW Core
- KAFKA Adapter Bus
- ETFD
- KAFKA Alarms Bus
- OpenOLT Adapter
- OpenONU Adapter
- *** Adapter
VOLTHA 2.x: RW Core

For each request, **Affinity Router** sends a message to 2 cores.

**API Handler** receives the message and decides whether to execute it or keep it in standby.

Depending on the request, it can be executed by:

- **Device Agent** Manages the physical device lifecycle
- **Logical Device Agent (Flow Decomposer)** Manages the logical switch lifecycle and maps the flows to the underlying topology

The **Adapter Proxy** acts as a shim layer to abstract the APIs while the **Inter Container Kafka Proxy** manages the request in an RPC fashion.
VOLTHA 2.x Stabilization Brigade

WE ARE HERE

Stabilization brigade focus

VOLTHA 2.x master

Test framework & tests

VOLTHA 2.1 stabilization branch

new features

bug fixes
SEBA: SDN Enabled Broadband Access

Virtual OLT manages and抽象s PON

Many functions (Closed Integrated Device):
- Subscriber tunnel termination (QinQ, PPPoe)
- Accounting
- Hierarchical QoS
- Lawful intercept (wiretap)
- Wholesale tunnel relay (e.g. L2TP)
- Multicast
- Routing

Backbone Network
SEBA today

Can we disaggregate and embed in SEBA POD?
SEBA with SD-BNG

Residence

Central Office

Virtual OLT manages and abstracts PON

Manages 1 or more switches

SD-BNG

VOLTHA

Trellis

BNG-c

OLT MAC

P4 based WhiteBox Switch with BNG-user plane

Backbone Network
Plan for BNG disaggregation in SEBA

- BNG user plane (BNG-u)
  Implement “in-fabric” using P4 and merchant silicon
- BNG control plane (BNG-c)
  Implement as an app running on top of ONOS

Initial focus on PPPoE based BNG

Acknowledgments:

- Deutsche Telekom: Initial P4 implementation of PPPoE-based BNG user plane
Proposed BNG-U functional distribution

**ASG SW (Barefoot Tofino)**
- DC chipset
- Implemented in P4
- Use P4Runtime to manage flow tables at runtime

**OLT (Broadcom Qumran)**
- Deep buffer chipset
- Which API to manage queues and hierarchical scheduling policies at runtime? BAL?

---

**UPSTREAM**

- Classification
- Policing
- Double-VLAN termination
- Anti-spoofing
- PPPoE term.
- Routing
- ACL
- Accounting
- Lawful intercept

**DOWNSTREAM**

- Lawful intercept
- Accounting
- H-QoS
- PPPoE aggr.
- Double-VLAN aggregation
- Mcast replication
- ACL
- Routing
- Classification
SEBA with SD-BNG

Network Edge Mediator (NEM)

VOLTHA apps

Trellis apps

BNG-control

SDN Controller - ONOS

BNG-c app on ONOS

VOLTHA

OLT

ONU

Embedded BNG

P4-SWITCH

STRATUM

STRATUM

SERVERS

PPPoE server

SEBA POD

Docker

K8s

Helm

P4Runtime, gNMI, gNOI

Upstream Routers

Integrate existing control software

1. Integrate existing control software
2. Func distribution
SD-BNG Integration with SEBA
Demo at ONF Connect

Kubernetes cluster (compute node)

NEM

SEBA Apps  Trellis  BNG-c Relay

ONOS

VOLTHA

PPPoE Server
Handles authentication, address assignment (IPCP), keep-alive, etc.

Subscriber with PPPoE client

ONU

OLT

Switch (P4)

Upstream router

Internet

Handles authentication, address assignment (IPCP), keep-alive, etc.
DEMO: Advancing SEBA with P₄, Stratum & NG-SDN

Want to learn more?
Visit the booth to see our demo
What’s missing / Next steps

• **Integration with NEM**
  - Push events related to subscriber management
  - FCAPS support
  - PPPoE based workflows (for TT and DT)

• **BNG-u improvements:**
  - Add Hierarchical QoS for downstream traffic
    - Plan is to use BCM Qumran chip inside OLT with BAL 3.1 API
  - Support service delivery protocols other than PPPoE
    - E.g. IPoE with DHCP-based subscriber address assignment
  - Missing P4 pipeline features -- help needed!
Thanks!

Questions?