ONOS Testing

ONOS Build 2017

Jon Hall - JHALL@ciena.com
Unit Testing

- **What is it?**
  - Test for a small unit of code
  - Test inputs and outputs of a function
  - Insures the code does what we said it would
  - Helps detect when changes break this contract

- **What are our coverage goals?**
  - New code should have at least 70% coverage but the goal for the entire project is 80+%

- **How to run:**
  - `buck build onos`
  - `buck test`

- **Run as part of Gerrit Validation**

- **Resources:**
  - [ONOS Wiki:Unit Test Guidelines](#)
GUI Unit Testing

● What is it?
  ○ Javascript Unit tests
  ○ Run in a headless browser

● How to run:
  ○ npm install
  ○ npm test

● Run as part of the Gerrit Validation

● Resources:
  ○ See onos/web/gui/src/main/webapp/tests/README.txt

Time to run: ~1 Minute
STC Testing - Integration

- **What is it?**
  - Scenario Test Coordinator (STC)
  - Short, simple system tests
  - Designed to quickly verify you didn’t break any major functionality
  - Modular tests can be composed to create new scenarios
  - Control ONOS through Python or Bash scripts
  - Uses Mininet for the dataplane

- **How to run:**
  - `stc smoke` or `stc scenario`

- **Can submit a scenario with bug reports to help developers reproduce the issue**

- **Resources:**
  - [ONOS wiki: Appendix G: Scenario Test Coordinator (STC)]](ONOS_wiki:Appendix_G:Scenario_TestCoordinator_(STC))
ONOS System Tests using TestON

● **What is it?**
  ○ Thorough system tests
  ○ Written in python
  ○ Controls ONOS through Rest or CLI

● **3 types of tests:**
  ○ Functionality
  ○ Longevity (CHO)
  ○ Performance and Scale (SCPF)

● **How to run:**
  ○ `./cli.py run FUNCintent`

● **Tests are run nightly on master**
  ○ Used to help validate new releases
ONOS System Tests using TestON

- **Resources:**
  - Github: opennetworkinglab/OnosSystemTest
  - ONOS Project Gerrit: OnosSystemTest
  - ONOS Wiki: System Testing Guide
  - ONOS Wiki: System Test plans and Results
Functionality Testing

- Focus on the ONOS “Core” and move out
- FUNC
  - Intents, Topology, Flows, Flow Objectives, Groups, etc.
- HA
  - Controller failures, Control Network failures, Dataplane failures
- Usecases
  - App specific system tests
    - SDNIP, Segment Routing, VPLS, etc...
Longevity Testing

- CHO - Continuous Hours of Operations
- Find memory leaks, or hard to reproduce defects
- Random or fixed sequence of events

Time to run: Days
Scale and Performance

- Focused on ONOS performance
  - Mock out third party actors (topology elements) with “Null providers”
- Break down a single event from an operator’s perspective into the individual ONOS events
- Helps us focus development efforts
- Measure performance at different ONOS cluster sizes
- Example: Mastership Failover when an ONOS node dies
  - Cluster detects failed ONOS node
  - Cluster sends Mastership Event
  - New Master sends OF Role Request to device

- Latency tests:
  - Topology Events, Intent Events, Flow, Host Add, Mastership Failover

- Throughput tests:
  - Intents, Flows, Flow Objectives

Time to run: Hours
Other tests

- **Delta - Security Testing**
  - Integrated in the past, being worked on by sec/perf brigade members

- **Docker**
  - Simple test that starts a cluster from the latest dockerhub images
  - Candidate for community contributions
How to get involved

● Different brigades:
  ○ Build and Package Infrastructure
  ○ GUI
  ○ Security and Performance Analysis

● Is your favorite app missing tests?
  ○ We can help you write tests
  ○ Even just test plans can be helpful if you can’t write code
  ○ Reach out to the brigade related to your app, or form a new one around it