Evaluation of ONOS performance

Open Networking Foundation

Goals

- We designed a set of experiments to characterize latencies, throughput and capacities of ONOS under various application and network environments.
  - Topology change latency
  - Topology scaling
  - Flow setup latency/throughput
  - Intent operations latency/throughput
  - Cbench (packet-in processing rate)

- By analyzing the results, we hope to
  - provide network operators and application developers with a "first look" of ONOS' performance capability.
  - In addition, the performance results should help developers gain insights for identifying performance bottlenecks and optimization opportunities.

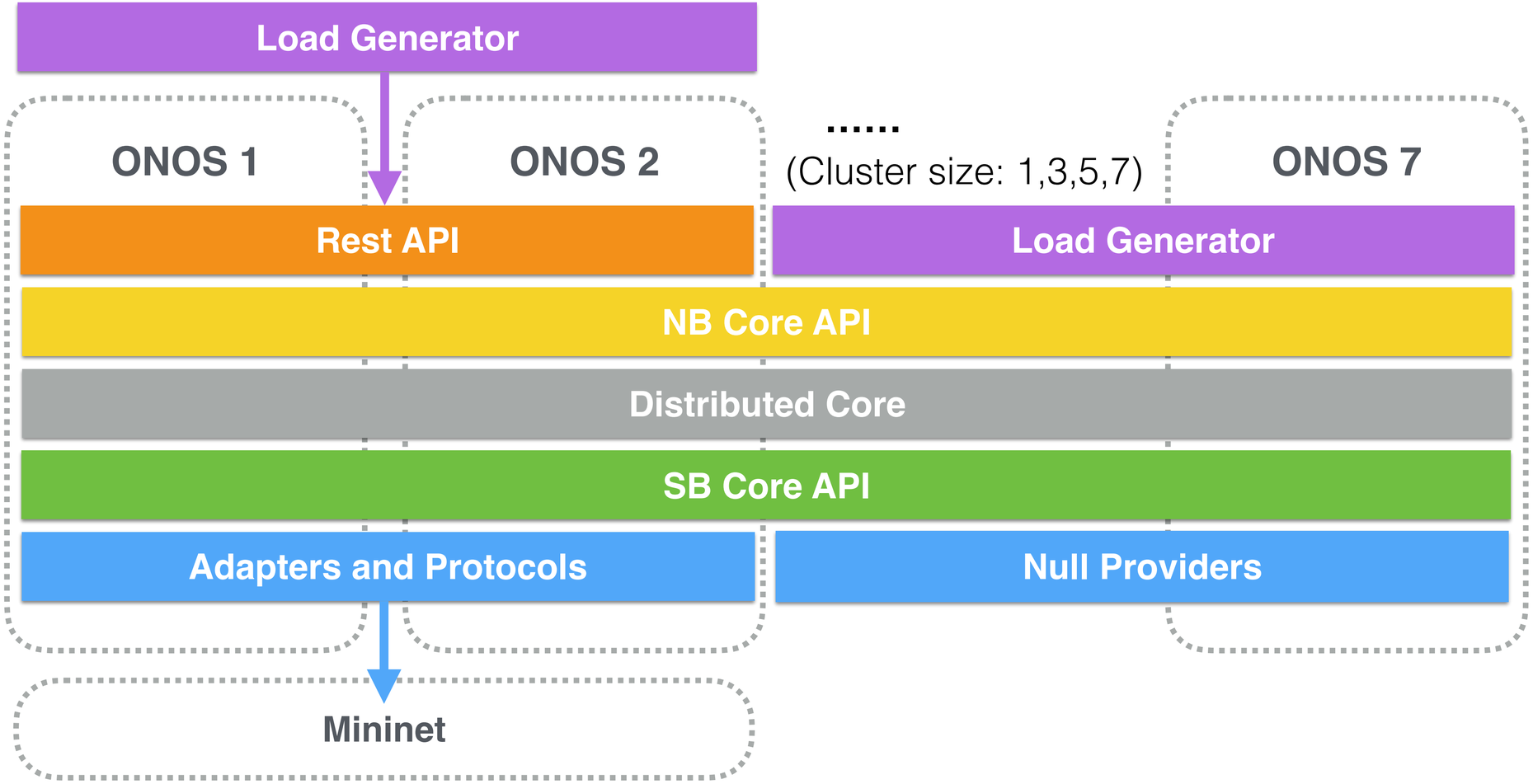
- Note: test results are from **onos-1.12** branch (comparing with onos-1.10)

Methodologies

- Performance measured at increasing scale
  - The general theme of all test cases is to make measurements on ONOS as it scales from 1 node to 3, 5, 7 nodes.

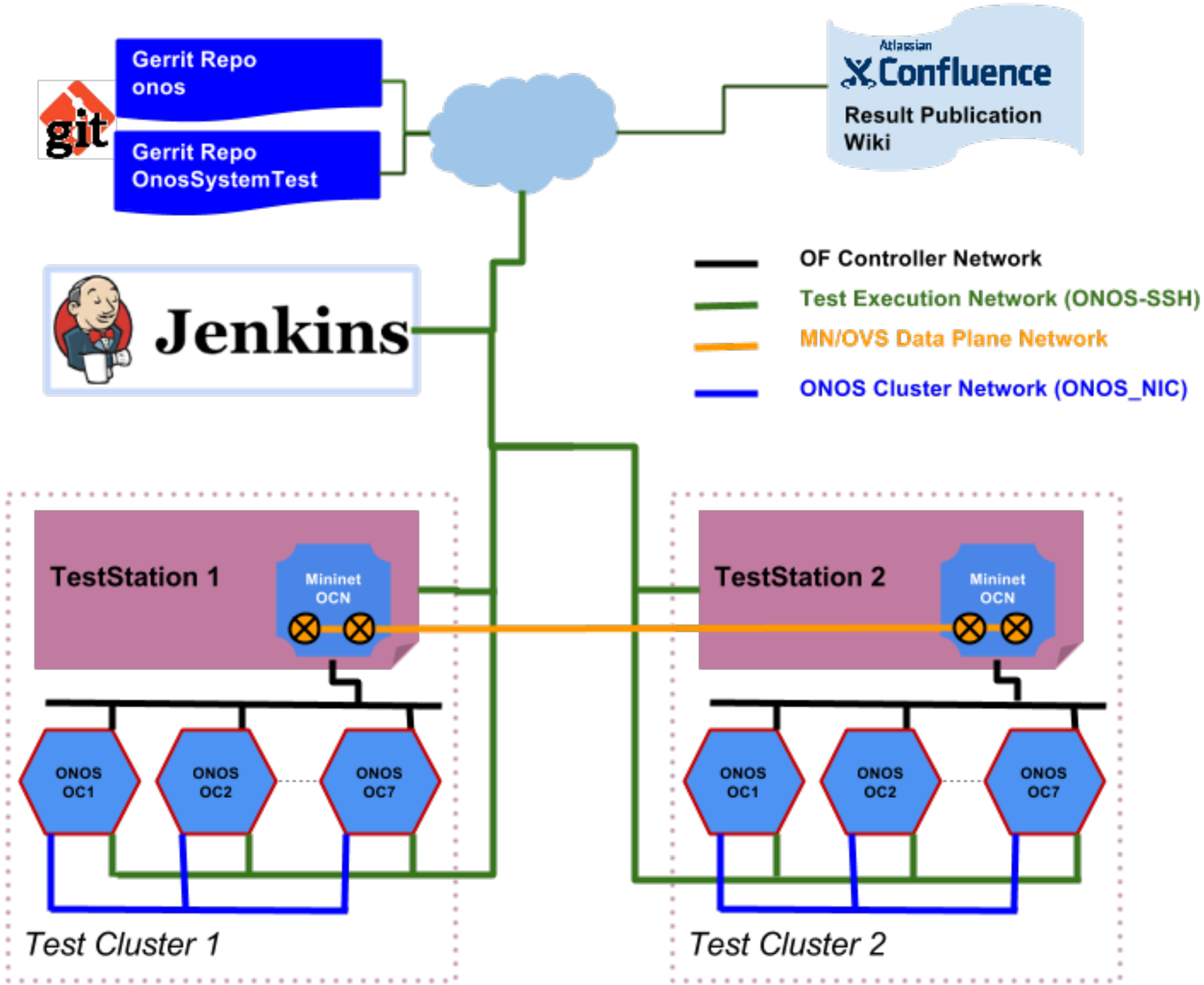
- In order to characterize ONOS' intrinsic characteristics we developed a few utilities for the experiments
  - **Null Providers** that act as device, link, host producers as well as a sink of flow rules.
    - for bypassing Openflow adapters and eliminate potential performance limits from having to use real or emulated Openflow devices.
  - **Load generators** that interface with ONOS Java APIs
    - for generating a high-level of loading from the application or the network interfaces to stretch ONOS's performance limits.
  - **Meters** in "topology-events-metrics" and "intents-events-metrics" apps
    - for some of the timing and rate related tests to capture key event timestamps and processing rates

Methodologies



Testing Environment

- Gerrit+Jenkins+Wiki
- Test clusters
  - Bare-metal Cluster:
    - 7 onos instances
    - TestON + Mininet

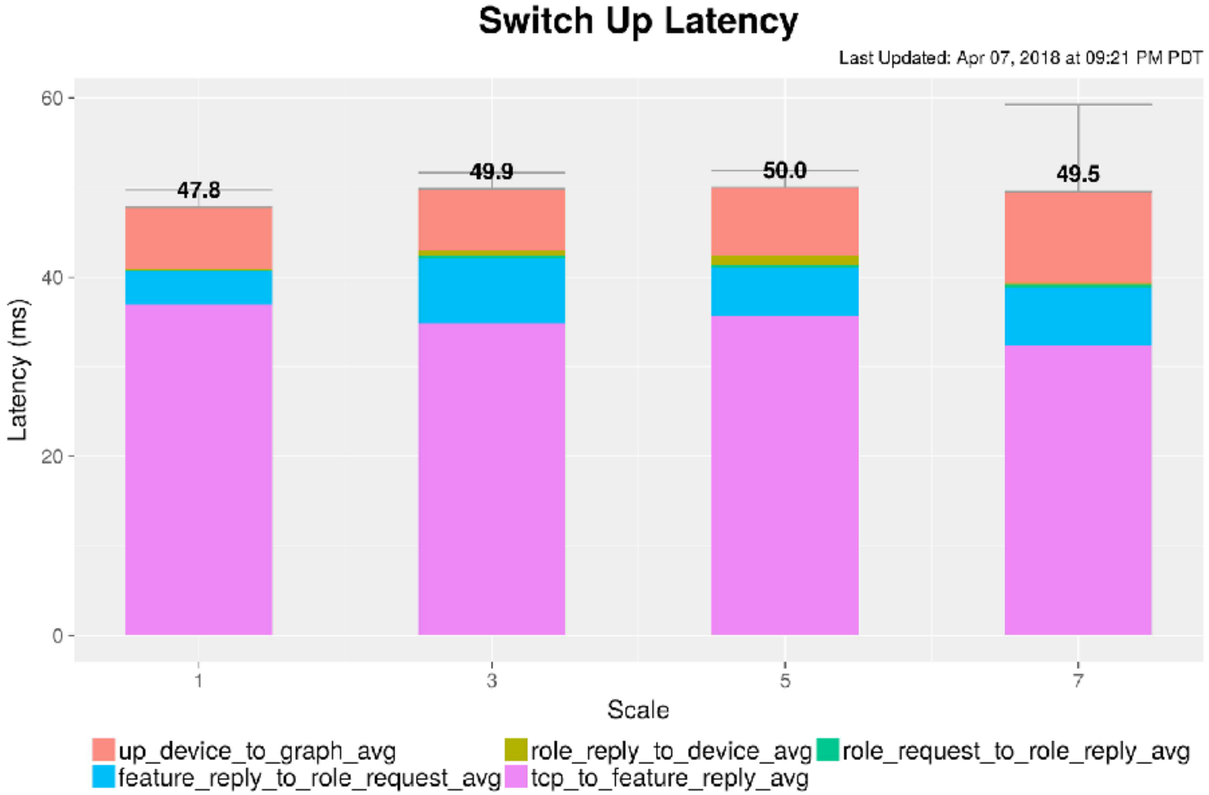


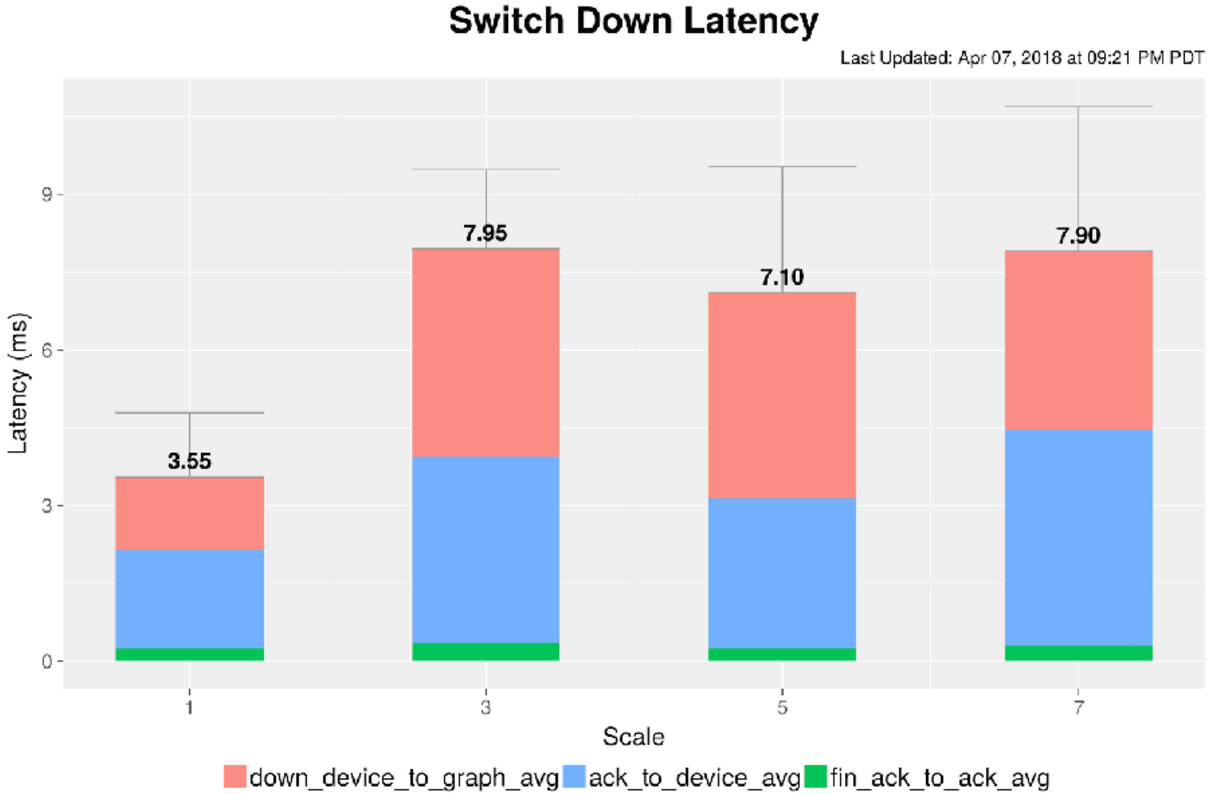
Topology Change Latency

- To measure how quickly ONOS can respond to different types of topology events, such as port up/down, switch add/remove or host discovery (tested with OpenFlow)
  - Switch up/down latency
  - Port up/down latency
  - Host discovery latency

Switch Up/Down Latency

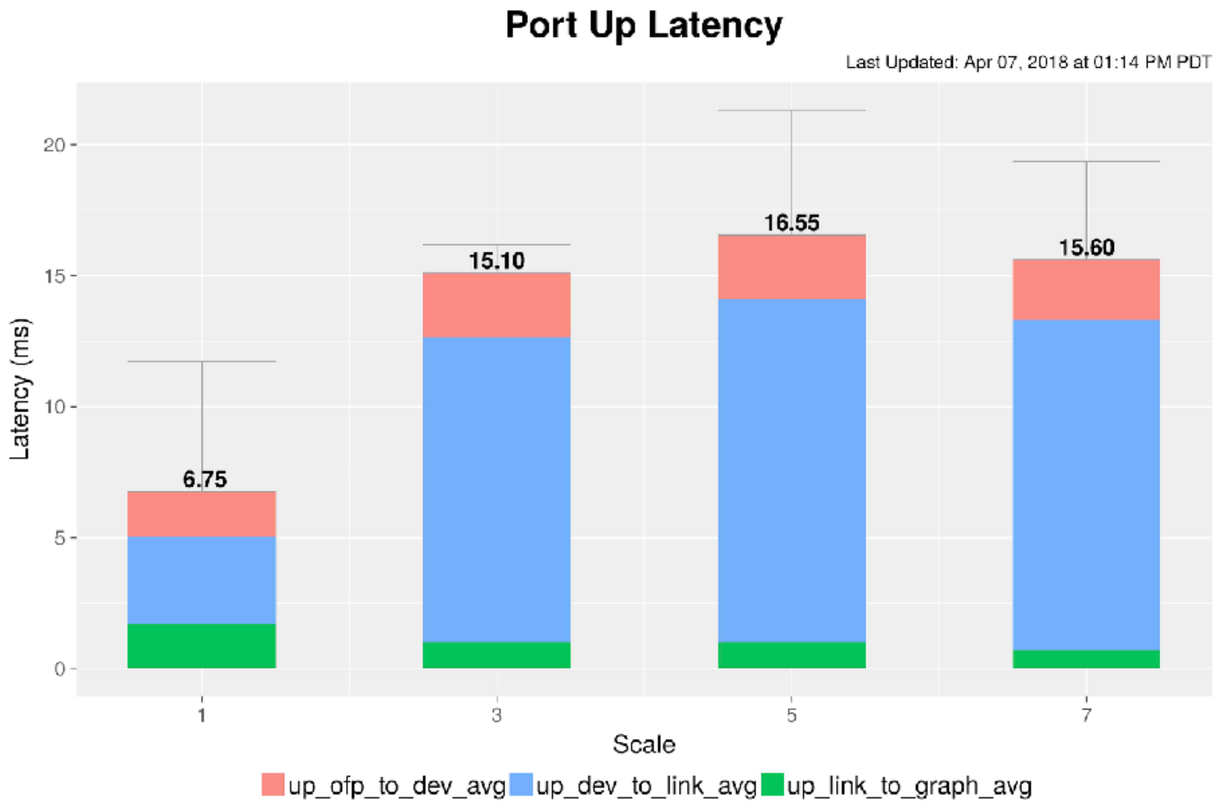
- Switch up takes **50ms** which is the same as onos-1.10
- Switch down takes **3ms** for single-instance and **7ms** for multi-instance (it was 5ms in onos-1.10 and the increase was due to a functionality fix)

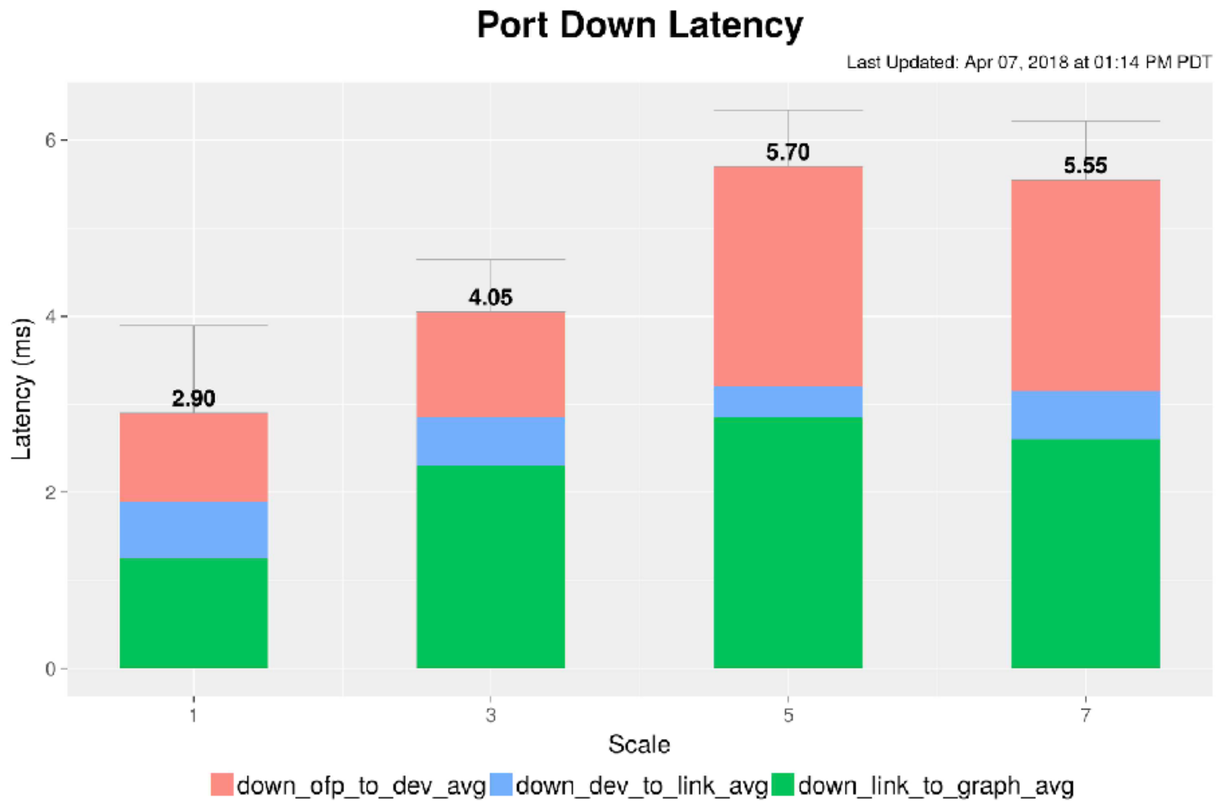




Port Up/Down Latency

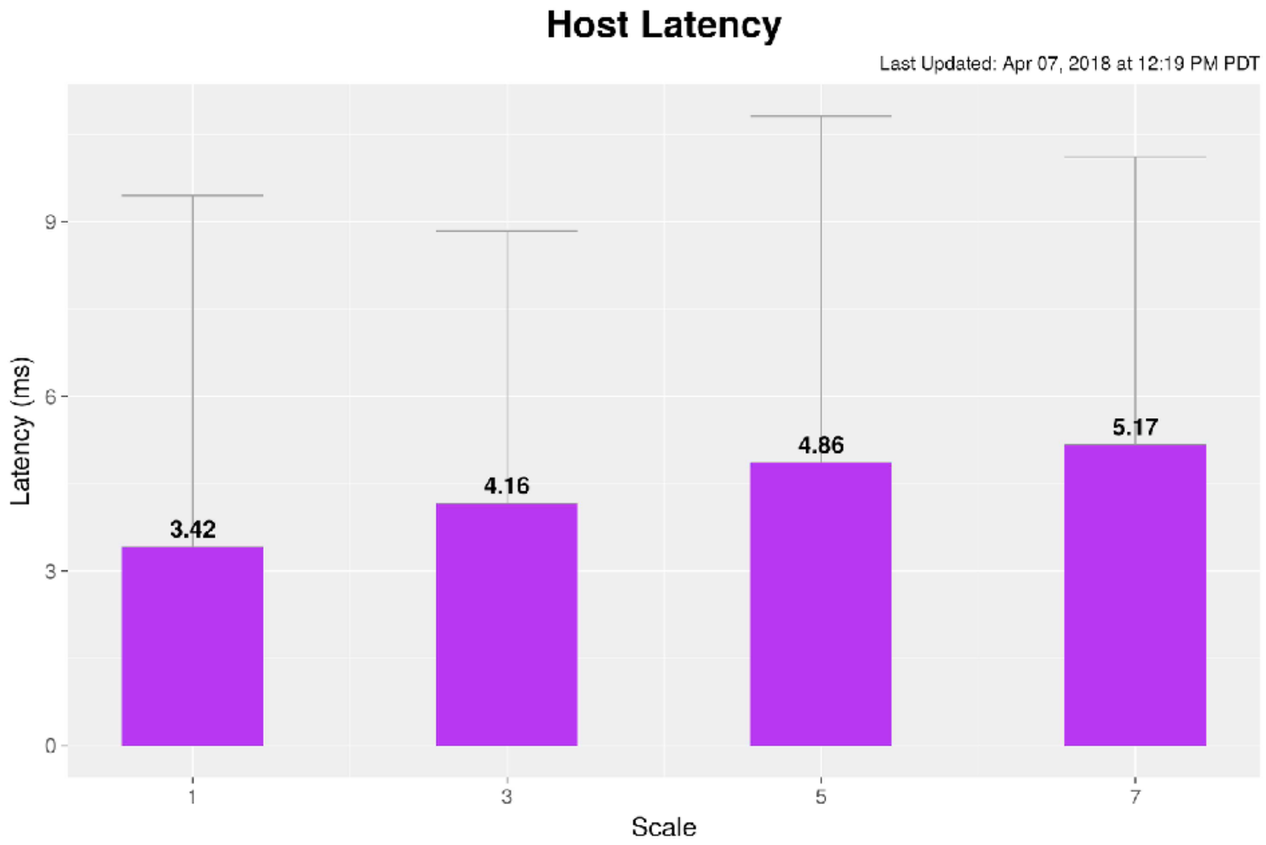
- Port up takes **7ms** for single-instance and **15ms** for multi-instance
- Port down takes **3~5ms**
- Results stay the same as onos-1.10





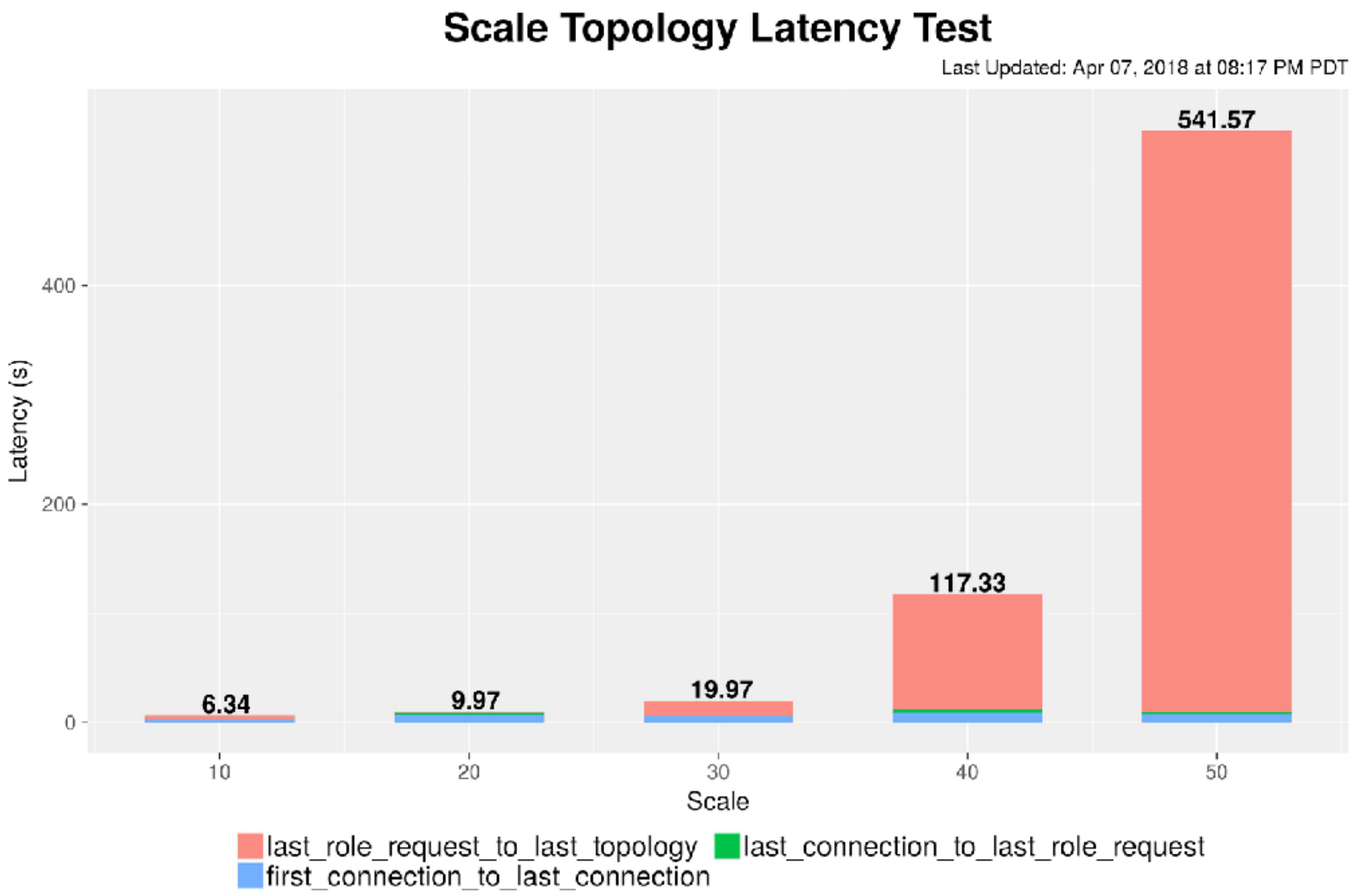
Host Discovery

- Host discovery latency is around **4ms**
- Latency in multi-instance case dropped from ~100ms (onos-1.10) to 4-5ms



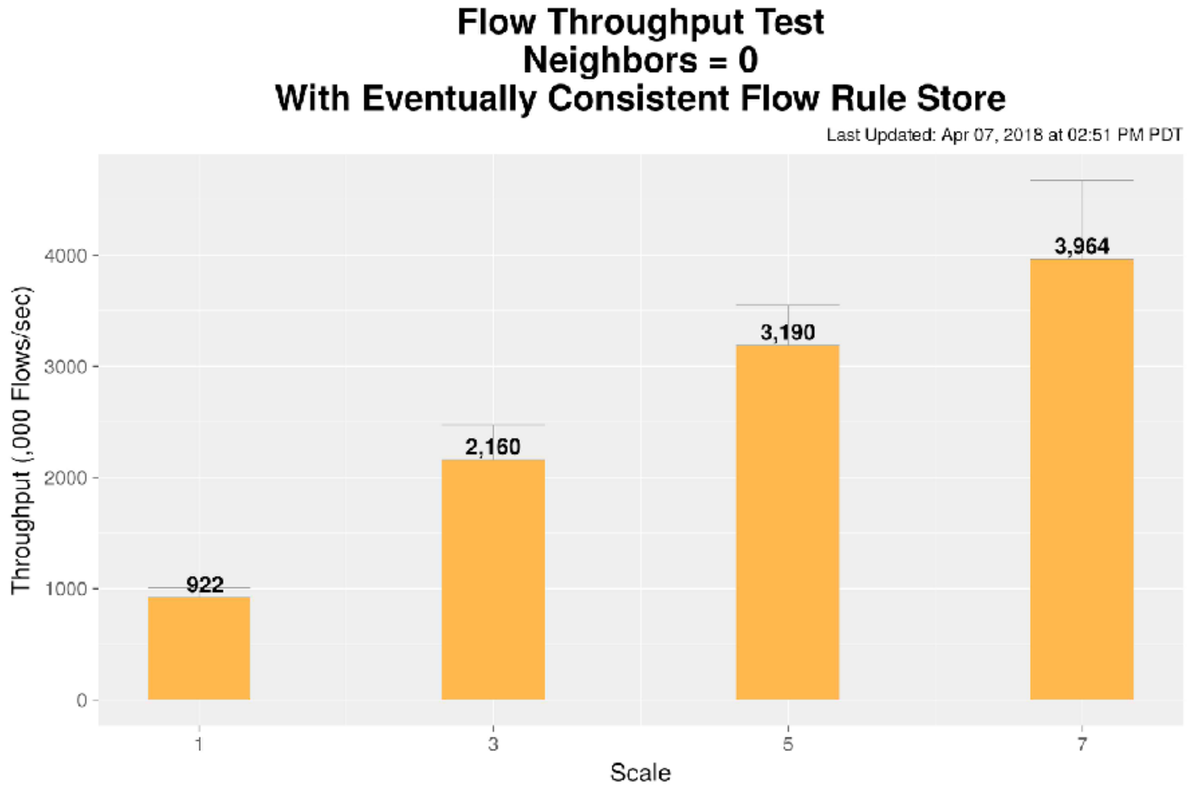
Topology Scaling

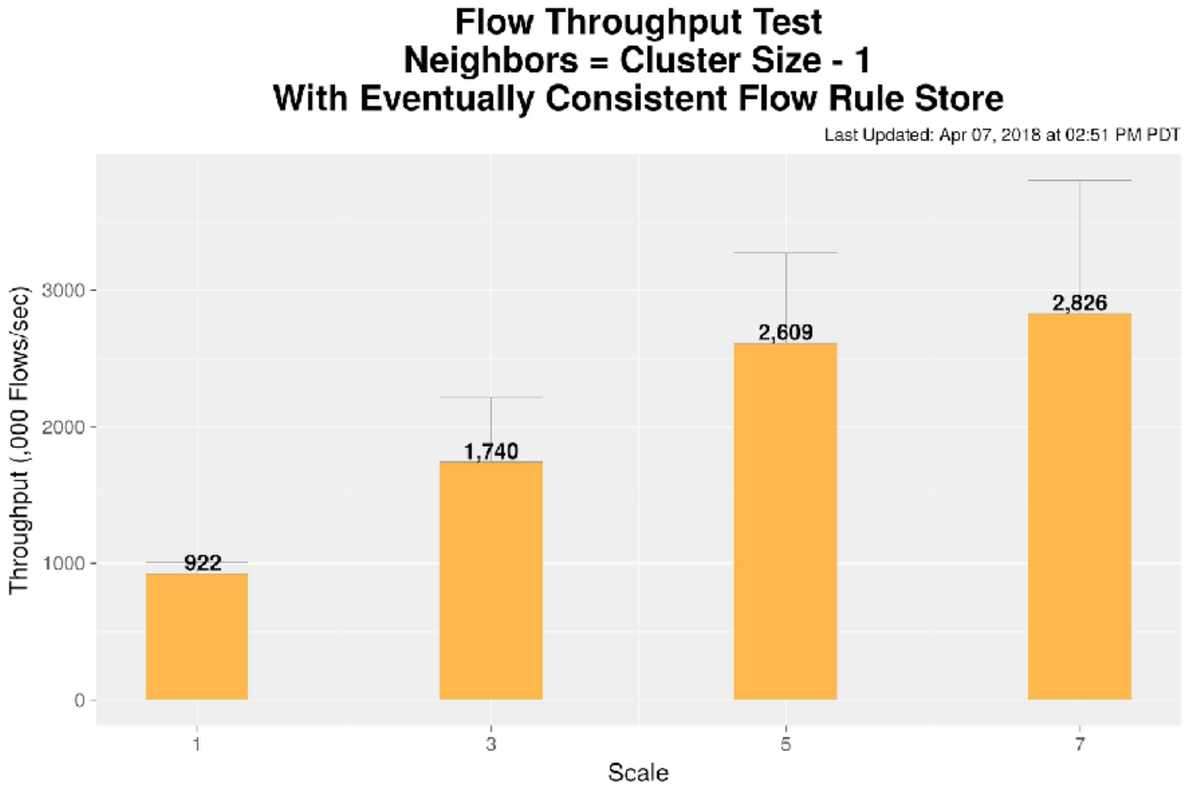
- To measure the latency and capacity for ONOS to discover and maintain the data plane topology
- Tested with OpenFlow
- A 3-node ONOS cluster can discover and maintain up to **50x50** switches with the same number of hosts in a torus topology in Mininet
  - It was 40x40 in ono-1.10



Flow Setup Throughput

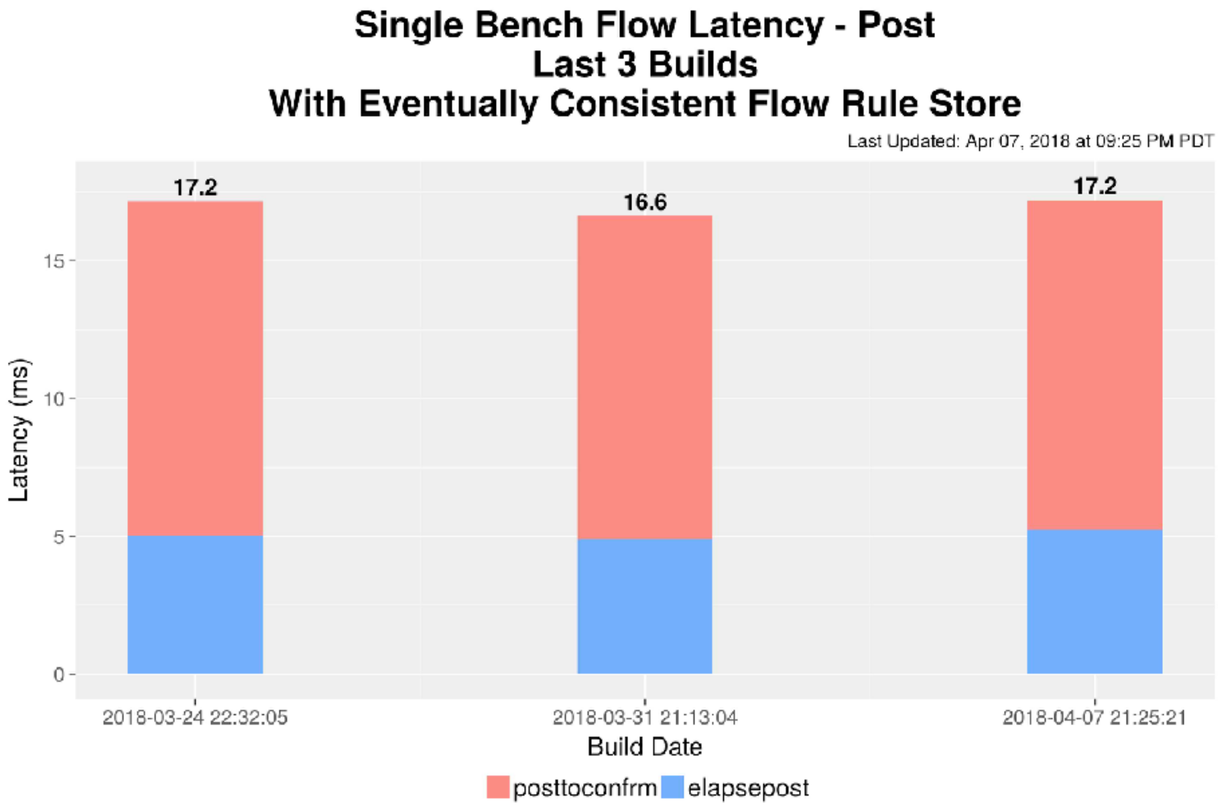
- To measure the ability of ONOS to handle an increasing number of flow setup requests, and the maximum load supported
  - Tested with load generator (Java API) and null-providers
  - Over **3million flows/s**. (Results stay the same as onos-1.10)
  - Note: Eventually Consistent flow rule store is being used by the flow rule subsystem

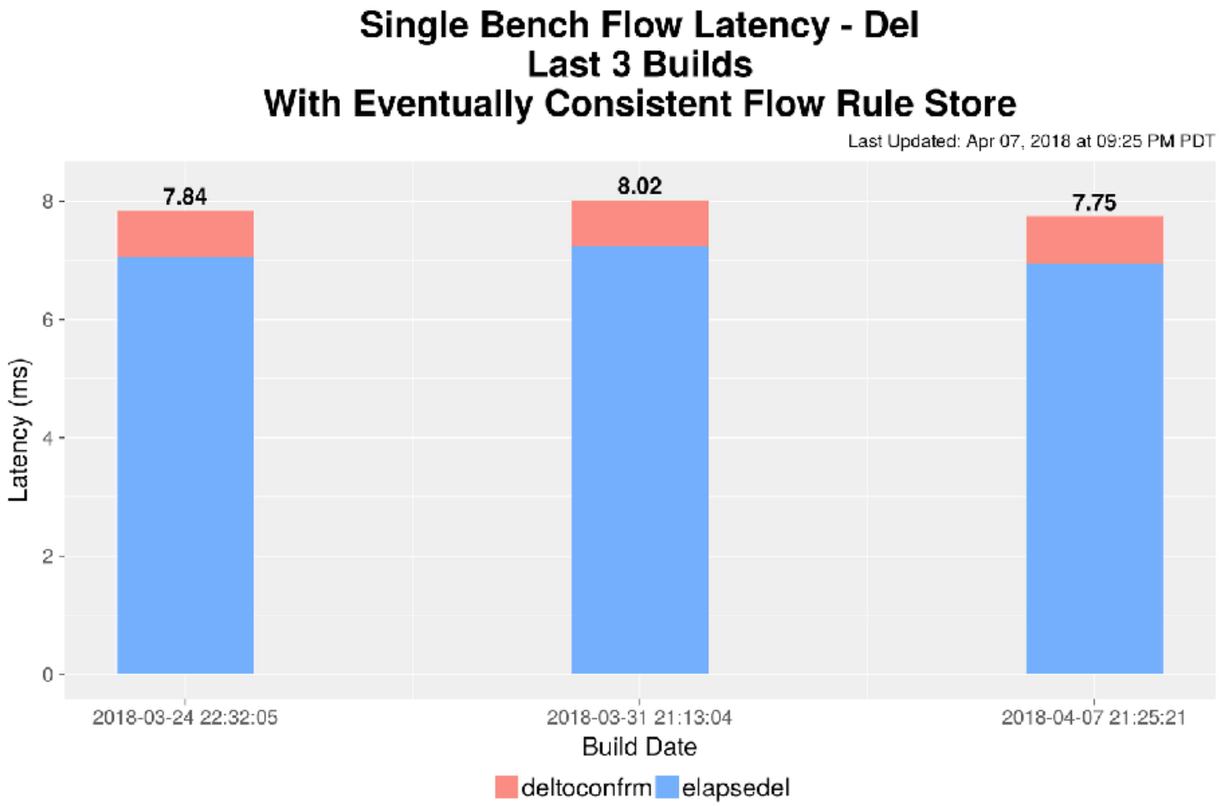




Flow Setup Latency

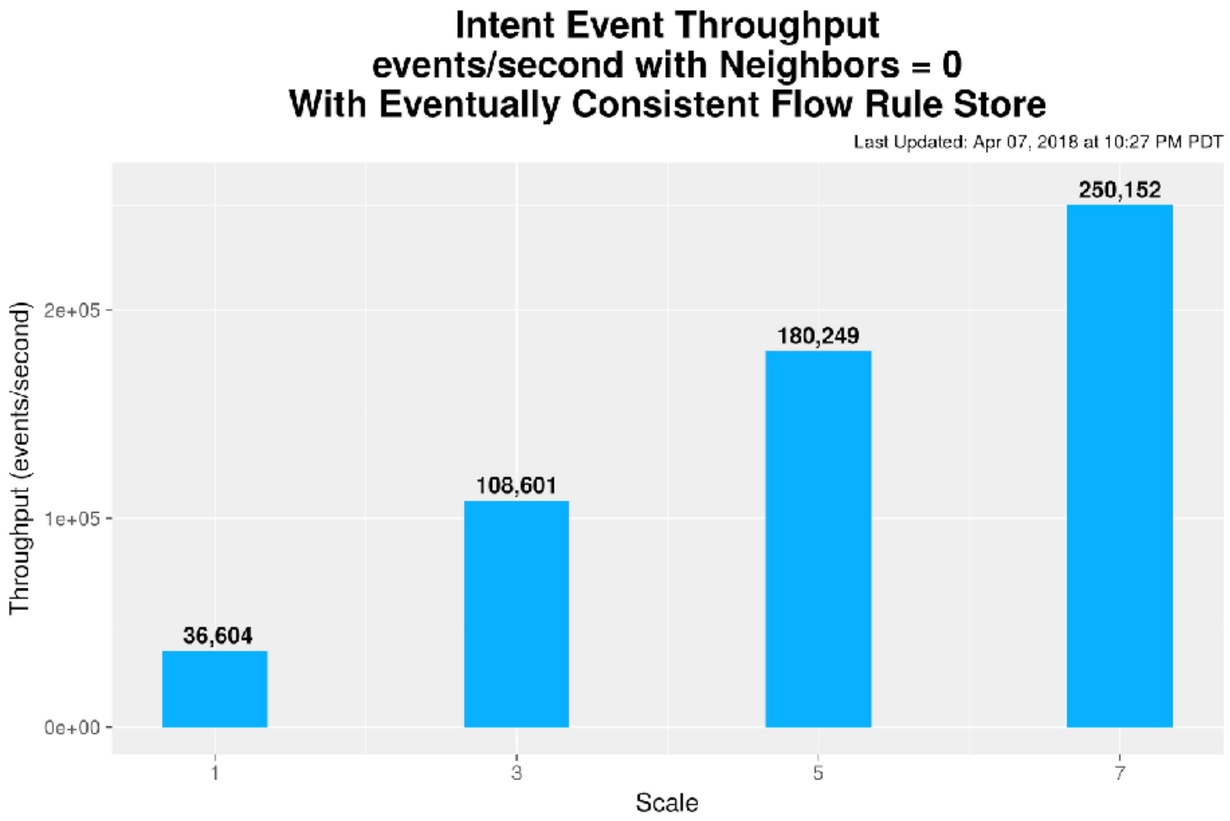
- To measure the latency of ONOS to install and remove flows via REST API.
  - 63 switches and 100k flows in 500 batches (Tested with OpenFlow)
  - Results stay the same as onos-1.10

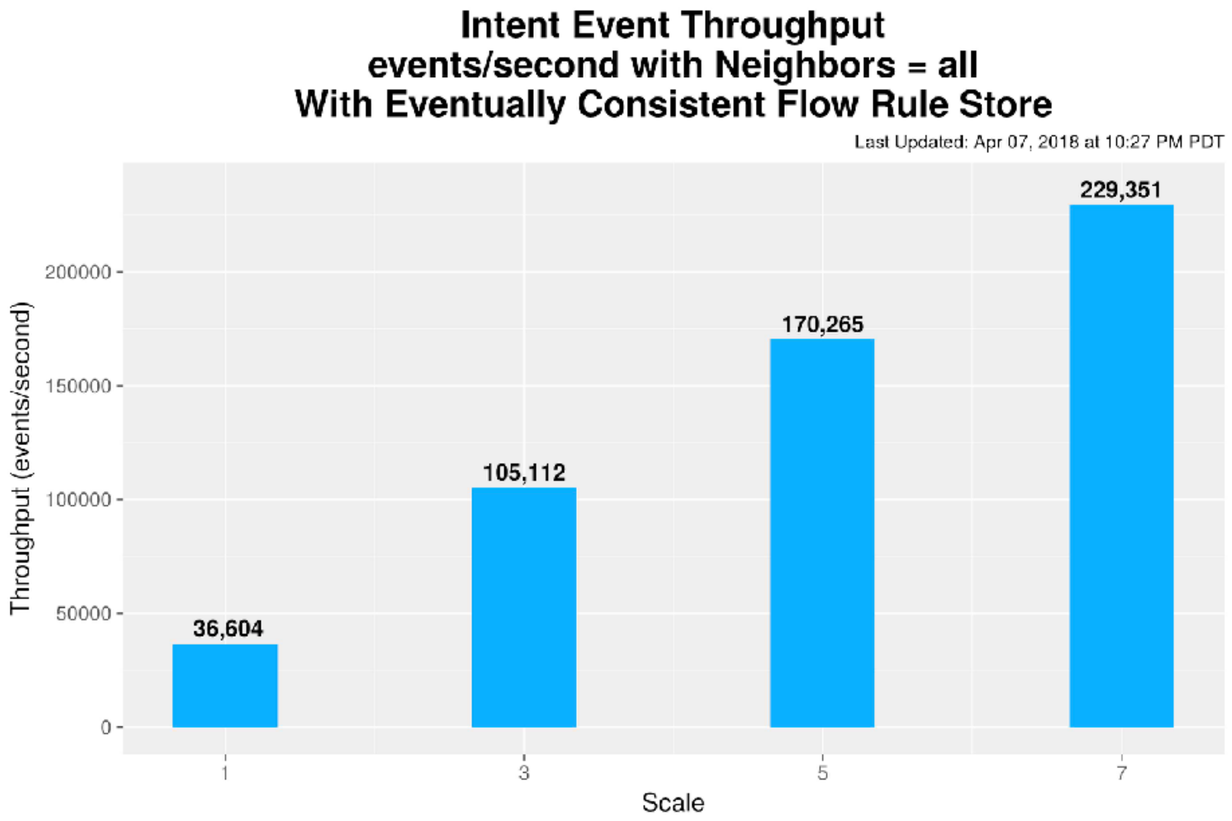




Intent Operations Throughput

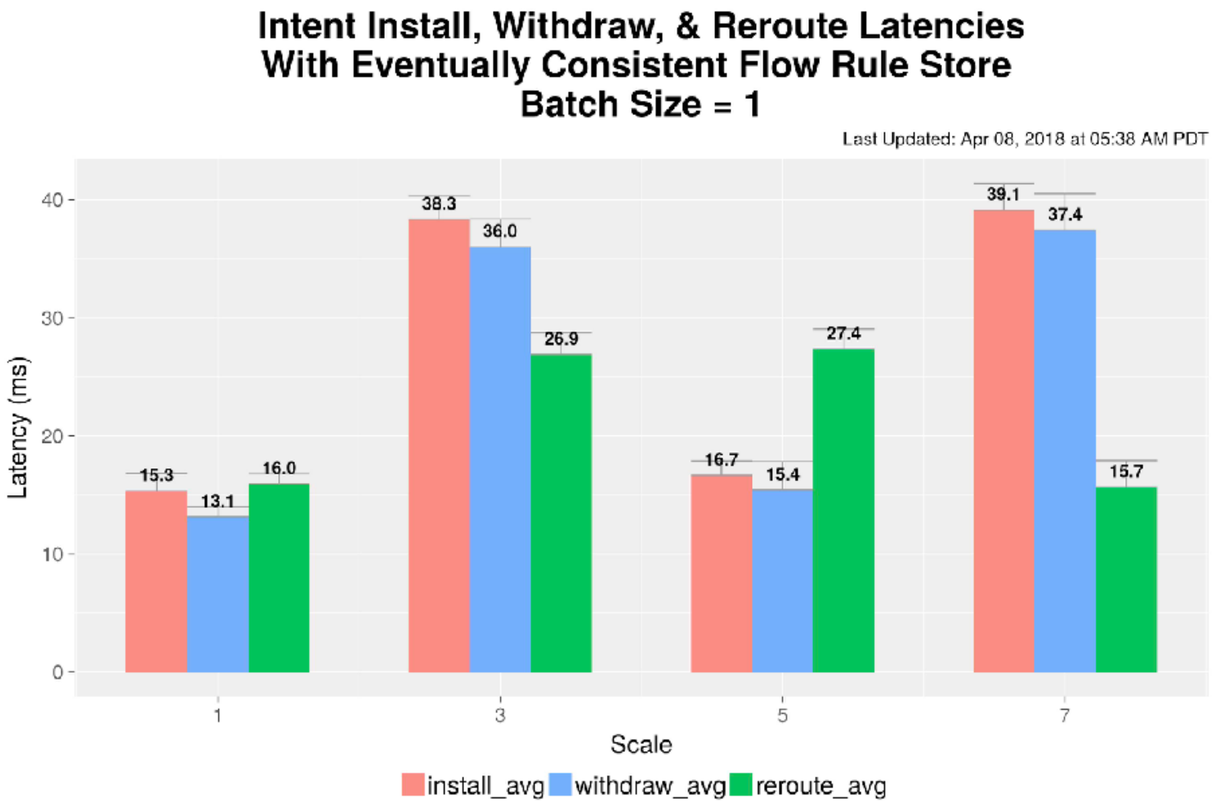
- To measure the ability of ONOS to handle an increasing number of intent requests, and the maximum load supported
  - Tested with load generator (Java API) and null-providers
  - Over **200k intents/s** (Results stay the same as onos-1.10)

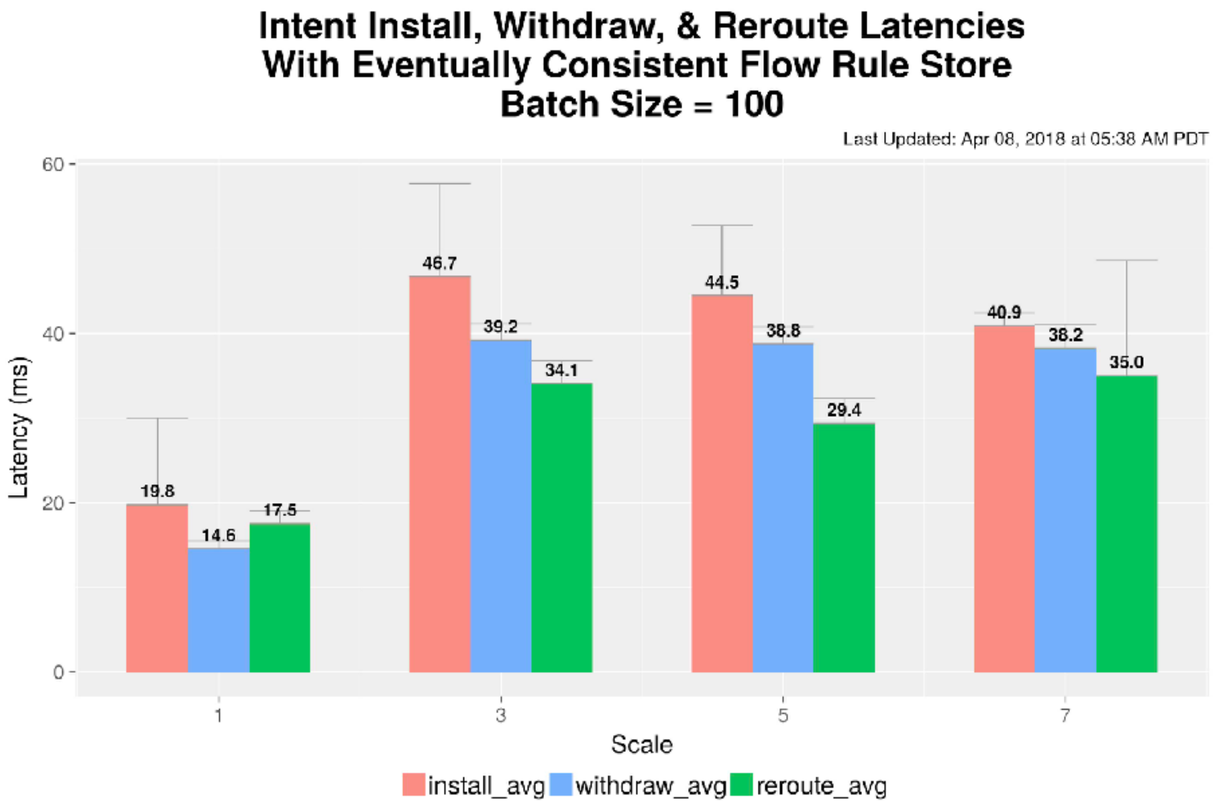




Intent Operations Latency

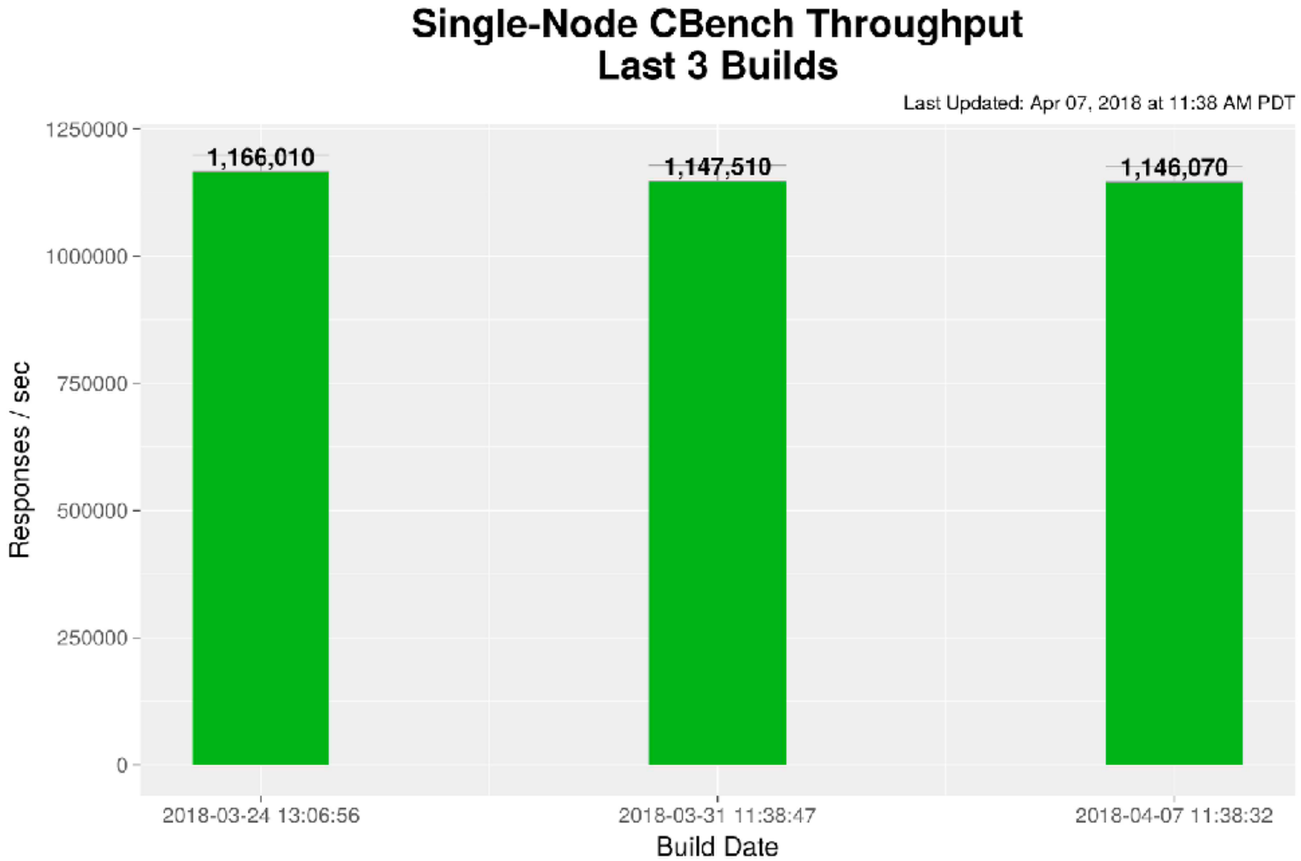
- To measure how quickly ONOS is able to satisfy an intent request and how quickly it can react to network failure events.
  - Tested with load generator (Java API) and null-providers
  - Results stay the same as onos-1.10





Cbench

- To measure how quickly ONOS processes and responds to packet-in messages
  - ONOS is able to process over **1million packets/s**
  - Result got increased from ~700k packets/s (onos-1.10)



More Information

- The tests are accomplished by leveraging the automation-testing framework called TestON.
  - It allows us to consistently setup a typical user environment and emulate their interactions with ONOS in a methodical and repetitive fashion.

- More information on Performance and Scale Tests
  - TestON Guide https://wiki.onosproject.org/display/ONOS/System+Testing+Guide
  - Test Plans https://wiki.onosproject.org/pages/viewpage.action?pageId=3441823
  - Test Results (1.12) https://wiki.onosproject.org/display/ONOS/1.12-Performance+and+Scale-out

Q & A

- Thanks!