SDN-IP Network with ONOS Controllers

Author: Roger Cortes

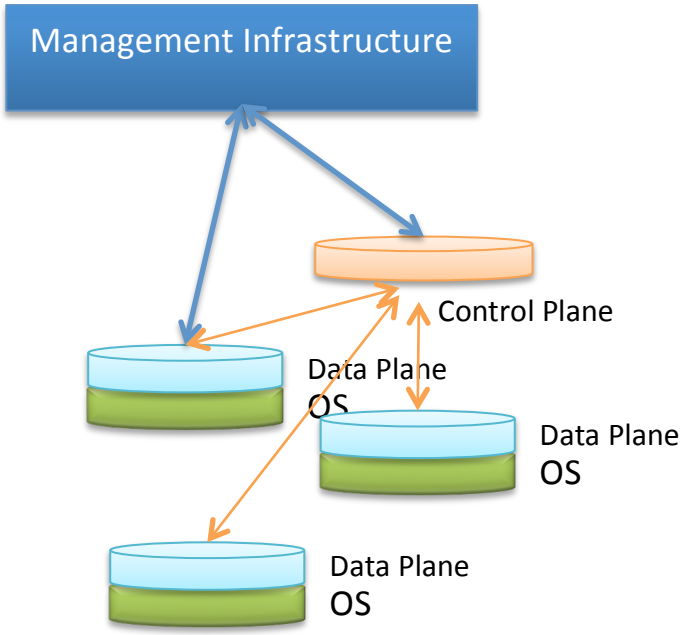
Background

This document serves as a guide in setting up and bringing up Agema Systems platforms in the SDN-IP network. The Agema Systems switches are loaded with Open Network Linux (ONL). Broadcom's OpenFlow Data Plane Abstraction (OFDPA) application is also installed in the switches. The control plane is separated out of the switches and moved onto the Open Network Operating System (ONOS) controllers. The platforms used in the setup are AG7648 switches, but any other Agema Systems platforms are supported.

Software Defined Networking (SDN)

Software-defined networking (SDN) is an umbrella term encompassing:

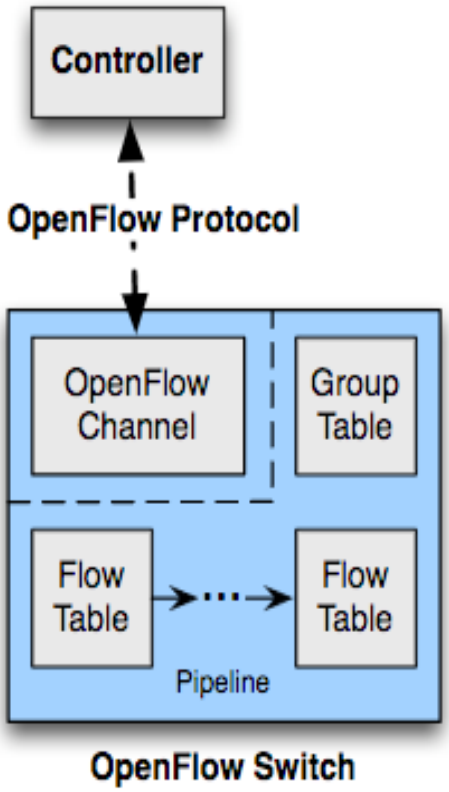
- Move of networking equipment from proprietary to standard CPU and a standard network processor unit (NPU)
- Separating control plane from data plane
- Making forwarding and flows programmable

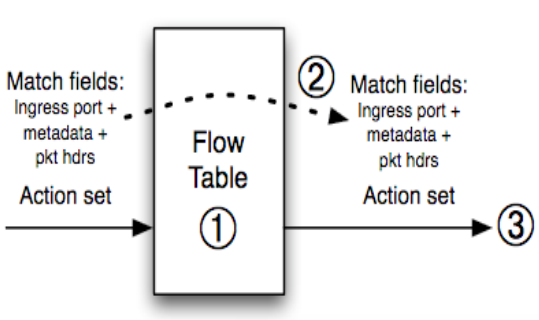


OpenFlow Data Plane Abstraction (OFDPA)

Following are the main components of OFDPA. For the latest specifications, refer to http://archive.openflow.org/wp/documents/.

1. **Controller**: to manage and push the flows
2. **OpenFlow protocol**: to exchange messages
3. **OpenFlow Channel**: interface that connects each OpenFlow switch to a controller
4. **Flow Table**: Table containing flow entries
5. **Flow Group**: Group of flows and their corresponding actions

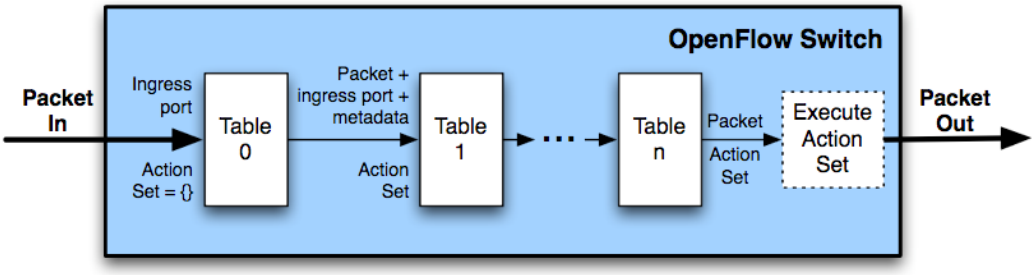




① Find highest-priority matching flow entry

② Apply instructions:
- i. Modify packet & update match fields (apply actions instruction)
- ii. Update action set (clear actions and/or write actions instructions)
- iii. Update metadata

③ Send match data and action set to next table



NOTE: Packets can be matched against multiple tables.

Main Components of Flow Entry in Table

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie

Example: Priority=500 in_port=50 action=groupId:65585 cookie=0x0,

Main Components of Group Entry in Table

Group Identifier	Group Type	Counters	Action Buckets

Example: groupId:65585 action=output:49

Main Components of Meter band Entry for QoS rates

Band Type	Rate	Counters	Type specific arguments

Example: meter=10 drop:rate=10000

Main Components of Meter Entry for QoS

Meter Identifier	Meter Bands	Counters

Example in_port=50 meter=10 action=output:49

Deployment Topology

In the topology below, the legacy BGP networks are simulated with the BGP routers. Quagga instances are used as BGP speakers, although any router would work.

Because ONOS Cluster talks to BGP speakers, it may be best to use Quagga running on VM as low-cost reflector to ONOS.

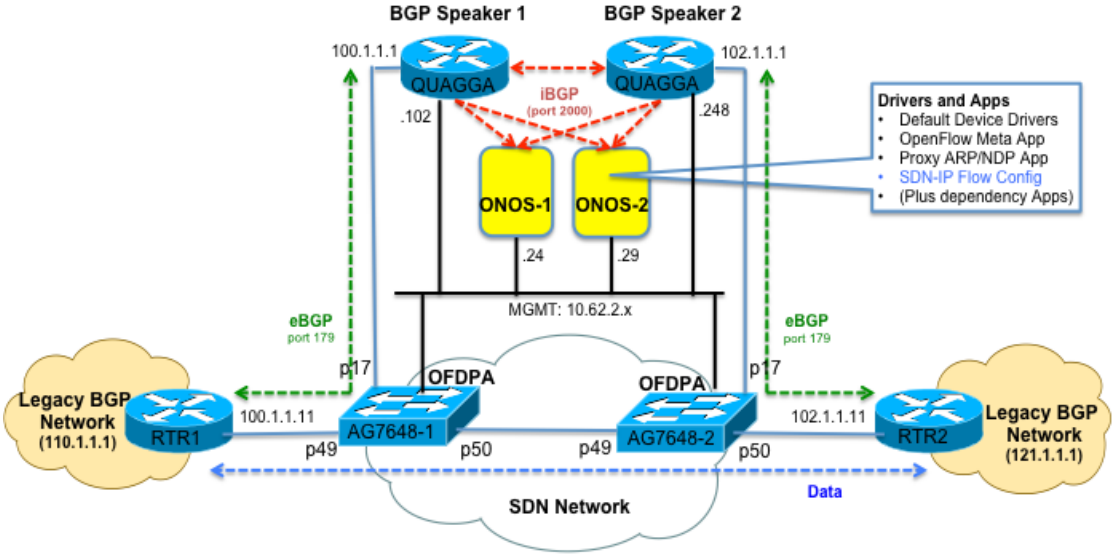
Required packages/applications

- ONL 2.0.0+
- OFDPA 3.0.3.1+
- Ubuntu 14.04+
- Apache Maven 3.3+
- Java 1.8+
- Quagga
- ONOS 1.10.2+
- Script to manually install flows into the OFDPA-enabled switches' flow table (sendFlow.sh) and JSON files.

Refer to **Reference** section (towards the end of this document) for packages/images download links.

Refer to **Appendix** section (at the end of this document) for a full list of script and JSON files used in this setup.

Minimum physical topology for the validation:



Routers and Switches Configurations

Here are key configurations for the **routers** and **switches** in the topology above.

AG7648-1 and AG7648-2

No static configurations except that the connected switchports must be enabled.

==========

RTR1

```
ip routing
exit
!
interface loopback 0
no shutdown
ip address 110.1.1.1 255.255.255.255
exit
!
! Connected to AG7648-1, port 49
interface 0/1
no shutdown
routing
ip address 100.1.1.11 255.255.255.0
exit
!
router bgp 100
bgp router-id 110.1.1.1
network 110.1.1.1 mask 255.255.255.255
network 100.1.1.0 mask 255.255.255.0
neighbor 100.1.1.1 remote-as 1000
neighbor 100.1.1.1 ebgp-multihop 255
address-family vpnv4 unicast
exit
```

==========

RTR2

```
ip routing
exit
```

```
!
interface loopback 0
no shutdown
ip address 121.1.1.1 255.255.255.255
exit
!
! Connected to AG7648-2, port 50
interface 0/3
no shutdown
routing
ip address 102.1.1.11 255.255.255.0
exit
!
router bgp 200
bgp router-id 121.1.1.1
network 102.1.1.0 mask 255.255.255.0
network 121.1.1.1 mask 255.255.255.255
neighbor 102.1.1.1 remote-as 1000
neighbor 102.1.1.1 ebgp-multihop 255
address-family vpnv4 unicast
exit
```

==========

BGP Speaker 1 (Quagga Router)

Zebra instance

```
hostname odl-zebra
password test
!
! Management port with IP address 10.62.2.102
interface eth0
```

```
 no link-detect
 ipv6 nd suppress-ra
!
! Connected to AG7648-1, port 17
interface eth1
 no link-detect
 ip address 100.1.1.1/24
 ipv6 nd suppress-ra
!
interface lo
 no link-detect
 ip address 21.1.1.1/24
!
ip forwarding
!
line vty
!
end
```

==========

BGP instance

```
hostname BGP-SPKR-1
password test
!
router bgp 1000
 bgp router-id 21.1.1.1
 ! IP address of ONOS-1
 neighbor 10.62.2.24 remote-as 1000
 neighbor 10.62.2.24 port 2000
 ! IP address of ONOS-2
 neighbor 10.62.2.29 remote-as 1000
```

```
 neighbor 10.62.2.29 port 2000
 ! IP address of BGP Speaker 2
 neighbor 10.62.2.248 remote-as 1000
 neighbor 100.1.1.11 remote-as 100
!
line vty
!
end
```

==========

BGP Speaker 2 (Quagga Router)

Zebra instance

```
hostname odl02-zebra
password test
!
interface lo
 no link-detect
 ip address 22.1.1.1/24
!
! Management port with IP address 10.62.2.248
interface p49p1
 no link-detect
 ipv6 nd suppress-ra
!
! Connected to AG7648-2, port 17
interface p50p1
 no link-detect
 ip address 102.1.1.1/24
 ipv6 nd suppress-ra
!
```

```
ip forwarding
!
line vty
!
end
```

==========

BGP instance

```
hostname BGP-SPKR-2
password test
!
router bgp 1000
  bgp router-id 22.1.1.1
  ! IP address of ONOS-1
  neighbor 10.62.2.24 remote-as 1000
  neighbor 10.62.2.24 port 2000
  ! IP address of ONOS-2
  neighbor 10.62.2.29 remote-as 1000
  neighbor 10.62.2.29 port 2000
  ! IP address of BGP Speaker 1
  neighbor 10.62.2.102 remote-as 1000
  neighbor 102.1.1.11 remote-as 200
!
line vty
!
end
```

ONOS Controllers Installation and Configurations

ONOS is installed on a PC or a VM running Ubuntu 14.04 or above. It is recommended to build a cluster of ONOS instances for load balancing and redundancy. In the test setup, a cluster of two ONOS instances interacts with the OFDPA-enabled switches. Here are key install instructions and configuration files for an **ONOS controller**.

ONOS Installation

This installation assumes that the home directory is */home/roger*.

Steps

1. Create *'opt'* directory.

   ~$ ***sudo mkdir opt***

2. Download and copy the *onos-1.10.2 tar.gz* package from *onosproject.org* wiki onto ~/opt directory.
3. Unpack and install ONOS.

   ~/opt$ ***sudo tar -zxf onos-1.10.2.tar.gz***

   ~/opt**$** ***sudo mv onos-1.10.2 onos***
4. Create *~/opt/onos/config* directory.

   ~/opt$ ***sudo mkdir onos/config***
5. Create *network-cfg.json* file and place it in *~/opt/onos/config* directory. Refer to *ONOS SDN-IP User Guide* for instructions.

Here's a sample network-cfg.json file used in the setup:

```
{
    "ports" : {
        "of:000000000000da7b/50" : {
            "interfaces" : [
                {
```

```
                    "name" : "AG7648-2",
                    "ips"   : [ "102.1.1.1/24" ],
                    "mac"   : "00:25:90:25:c7:2d"
                }
            ]
        },
        "of:000000000000da7a/49" : {
            "interfaces" : [
                {
                    "name" : "AG7648-1",
                    "ips"   : [ "100.1.1.1/24" ],
                    "mac"   : "00:25:90:25:77:65"
                }
            ]
        }
    },
    "apps" : {
        "org.onosproject.router" : {
            "bgp" : {
                "bgpSpeakers" : [
                    {
                        "name" : "bgp-spkr-2",
                        "connectPoint" : "of:000000000000da7b/17",
                        "peers" : [
                            "102.1.1.11"
                        ]
                    },
                    {
                        "name" : "bgp-spkr-1",
                        "connectPoint" : "of:000000000000da7a/17",
                        "peers" : [
                            "100.1.1.11"
```

```
                    ]
                }
            ]
        }
    }
}
```

6. Create *org.onosproject.routing.bgp.BgpSessionManager.cfg* file and place it in *~/opt/onos/apache-karaf-3.0.8/etc* directory.

Here's a sample org.onosproject.routing.bgp.BgpSessionManager.cfg file used in the setup:

```
bgpPort=2000
```

7. Repeat the above steps for the second ONOS instance. The *network-cfg.json* and *org.onosproject.routing.bgp.BgpSessionManager.cfg* are exactly the same.

SDN-IP Flows and Groups Install Script and JSON Files

The ONOS SDN-IP App isn't compatible with OFDPA at the time of testing, hence the intents-based flows pushed by the ONOS controller are rejected by the switch. To work-around this limitation, flows and groups have to be manually installed into the switch's flow table.

The required script and JSON files for manual installation of flows and group entries can be found at the **Appendix** section of this document. Here are the steps to prepare such files:

a) The following have to be created as separate files and placed in *'~/opt/onos-ofdpa/switch1/'* directory in the ONOS-1 controller.

aclFlow_port1749_icmp.json	aclFlow_port5017_tcp.json
aclFlow_port1749_tcp.json	aclFlow_port501.json
aclFlow_port1750_icmp.json	aclFlow_port502.json
aclFlow_port1750_tcp.json	l2IntGrp_port17.json
aclFlow_port4917_icmp.json	l2IntGrp_port49.json
aclFlow_port4917_tcp.json	l2IntGrp_port50.json
aclFlow_port492.json	l2RwGrp_port49.json
aclFlow_port493.json	l2RwGrp_port50.json
aclFlow_port5017_icmp.json	sendFlow.sh

b) The files shown above are hard-coded with values based on the AG7648-1's device ID, the controller IP address, source and destination IP/MAC addresses of the traffic between the legacy routers and BGP speakers, the ingress/egress switchports, etc.

c) Make a copy of all the files above for AG7648-2. Put them into *'~/opt/onos-ofdpa/switch2/'* directory.

d) Edit the sendFlow.sh file in *'~/opt/onos-ofdpa/switch2/'* directory to have the following:
switch_id=of:00:00:00:00:00:00:da:7b
switch_id_url=of%3A000000000000da7b

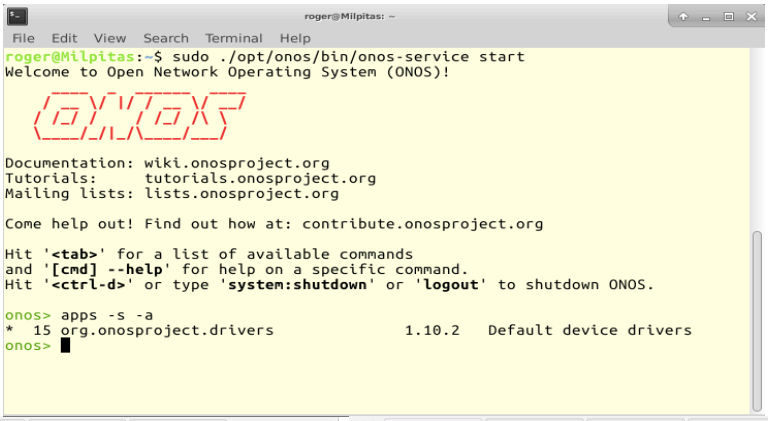
e) Edit the *aclFlow_*.json* files in *'~/opt/onos-ofdpa/switch2/'* directory to have *'deviceId=of:000000000000da7b'*.

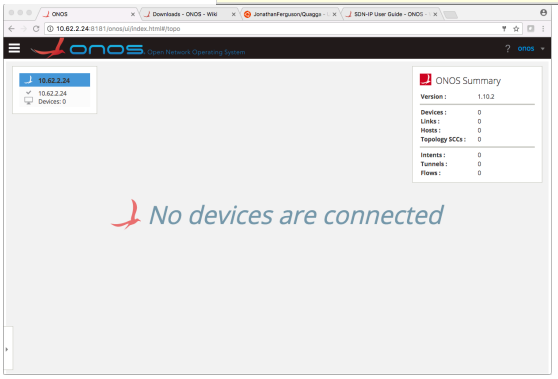
f) Copy *'~/opt/onos-ofdpa/switch1/'* and *'~/opt/onos-ofdpa/switch2/'* into ONOS-2 controller.

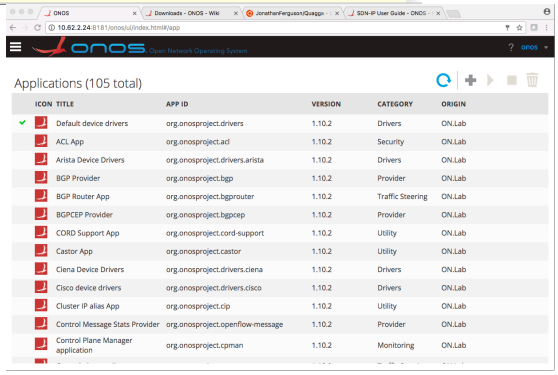
g) On ONOS-2, update the sendFlow.sh script with '*of_controller=10.62.2.29*' in both *'~/opt/onos-ofdpa/switch1/'* and *'~/opt/onos-ofdpa/switch2/'*.
'*10.62.2.29*' is the IP address on ONOS-2.

Test Execution and Results

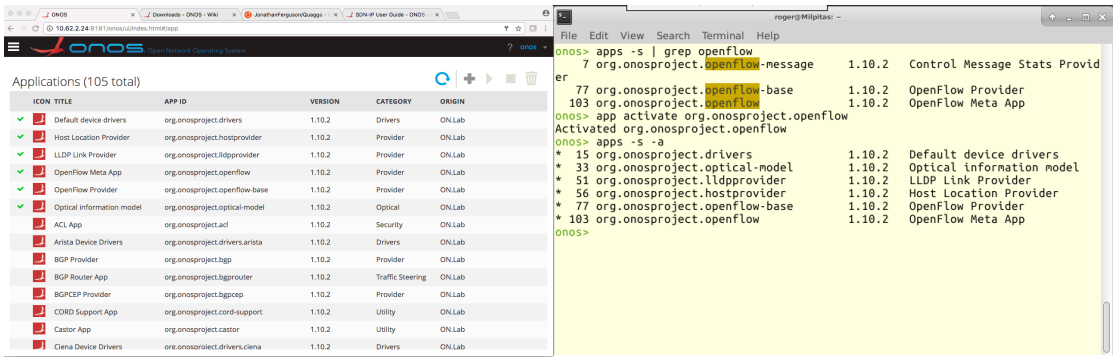
1. On ONOS-1, start ONOS and display the default active apps thru CLI and GUI (*http://10.62.2.24:8181/onos/ui/index.html* username/password: ***onos/rocks***).



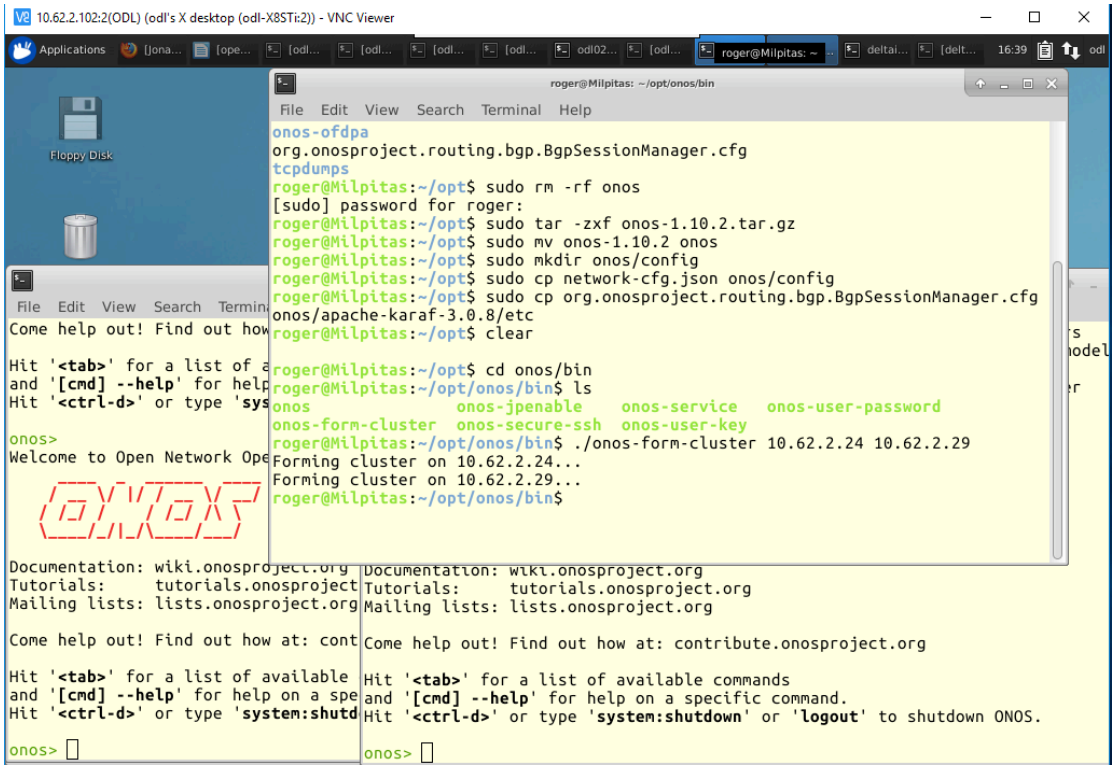




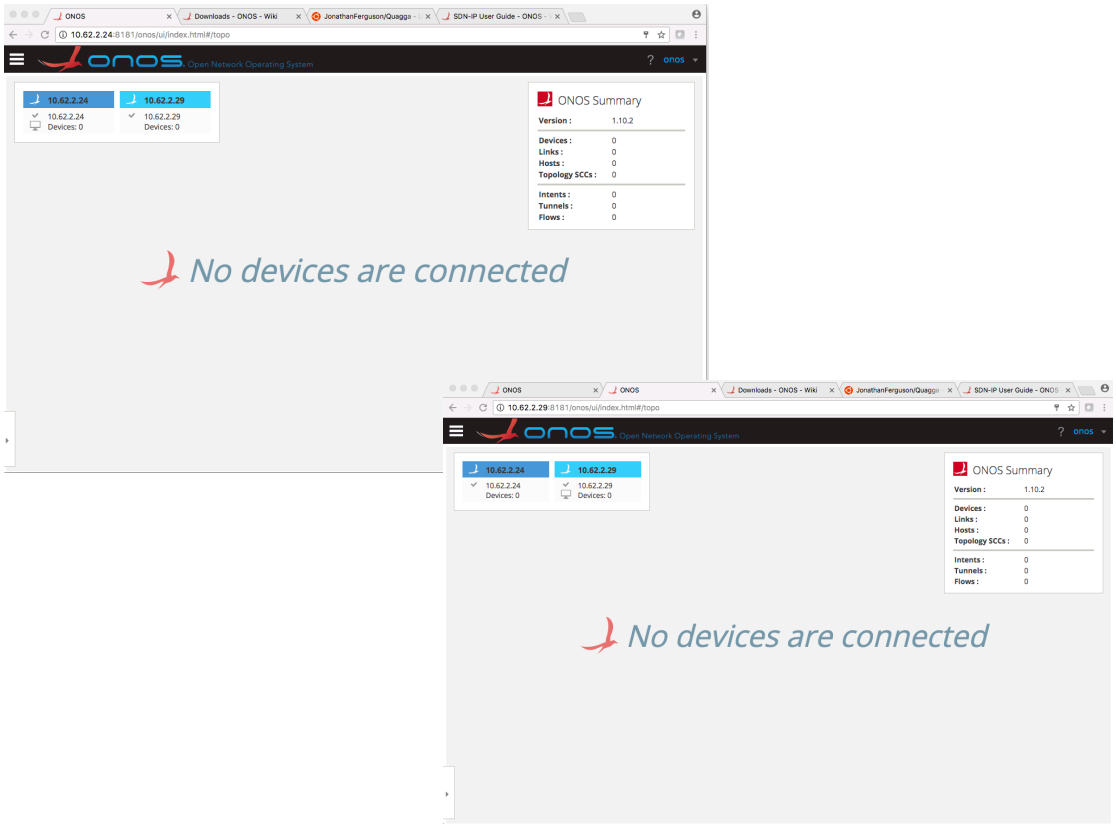
2. Activate OpenFlow and observe that the dependency apps get activated as well. Display the active apps via CLI and on GUI.



3. Start ONOS-2 and then form a cluster.
   a) Two ONOS instances (10.62.2.24 and 10.62.2.29) are running and they belong to a cluster.



b) The UI of 10.62.2.24 shows both ONOS instances. The UI of 10.62.2.29 also shows both ONOS instances.



4. Download and install OFDPA on AG7648-1 and AG7648-2.
   a) Refer to the References section of this document (below) for download location.
   b) After the package is downloaded onto the switches (ofdpa-ag7448_0.0.0.0.0_amd64.deb was used in the test), execute '*dpkg -i ofdpa-ag7648_0.0.0.0.0_amd64.deb*' to install the application.
5. On AG7648-1 and AG7648-2, login with username '*root*' and password '*onl*'.
6. Change working directory to */broadcom/ofdpa*.
7. Start *ofagentapp* on both AG7648-1 and AG7648-2. Both devices would need to connect to both the ONOS controllers (10.62.2.24 and 10.62.2.29) over TCP port 6633. The controllers also listen to TCP port 6653 if desired.

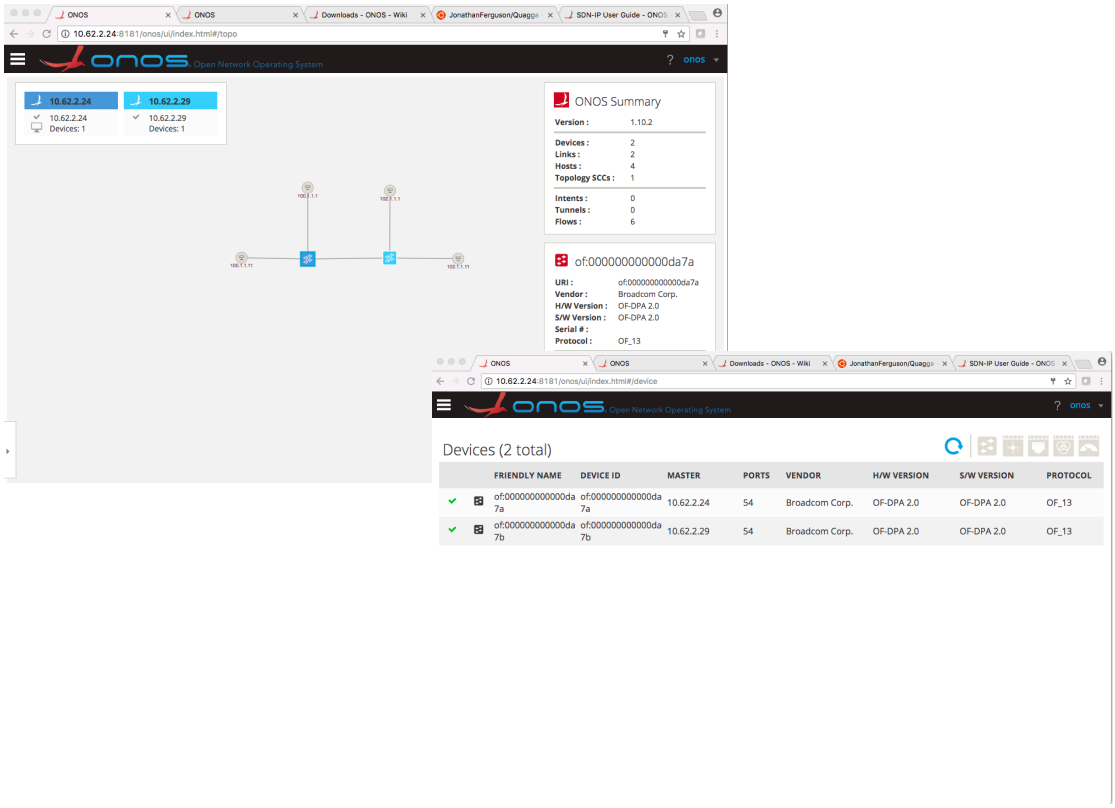
On AG7648-1 prompt, enter *'./launcher ofagentapp -t 10.62.2.24:6633 -t 10.62.2.29:6633 -i 0x000000000000DA7A &'*.

Device ID **000000000000DA7A** is the default ID used by OFDPA.

On AG7648-2 prompt, enter *'./launcher ofagentapp -t 10.62.2.24:6633 -t 10.62.2.29:6633 -i 0x000000000000DA7B &'*.

Device ID **000000000000DA7B** is used here, but any other value can work as long as it is unique so that this switch won't collide with AG7648-1 when connecting to the controllers.

8. Both the switches and the hosts (Legacy BGP routers and BGP Speakers) appear on the controller. Mastership is elected when the OFDPA switches connect to the controllers. In the screenshots below, AG7648-1 chose ONOS-1 (10.62.2.24) as the master while AG7648-2 chose ONOS-2 (10.62.2.29) as the master. However, it's also possible that both switches would elect the same controller as master.



9. Both switches should now have 3 flows each (LLDP, BDDP, and ARP flows). These are default flows installed by the controller.

   On AG7648-1:

   root@localhost:/broadcom/ofdpa# **./client_flowtable_dump**

   Table ID 60 (ACL Policy): Retrieving all entries. Max entries = 3072, Current entries = 3.

   -- inPort:mask = 0:0x0 srcMac:mask = 0000.0000.0000:0000.0000.0000 destMac:mask = 0000.0000.0000:0000.0000.0000 etherType:mask = 0x0806:0xffff | GoTo = 0 (None) outPort = CONTROLLER (Reserved) clearAction | priority = 40000 hard_time = 0 idle_time = 0 cookie = 10

   -- inPort:mask = 0:0x0 srcMac:mask = 0000.0000.0000:0000.0000.0000 destMac:mask = 0000.0000.0000:0000.0000.0000 etherType:mask = 0x88cc:0xffff | GoTo = 0 (None) outPort = CONTROLLER (Reserved) clearAction | priority = 40000 hard_time = 0 idle_time = 0 cookie = 12

-- inPort:mask = 0:0x0 srcMac:mask = 0000.0000.0000:0000.0000.0000 destMac:mask = 0000.0000.0000:0000.0000.0000 etherType:mask = 0x8942:0xffff | GoTo = 0 (None) outPort = CONTROLLER (Reserved) clearAction | priority = 40000 hard_time = 0 idle_time = 0 cookie = 7

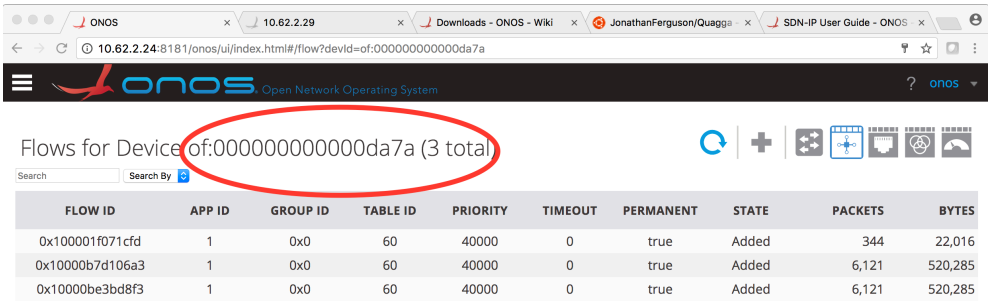
root@localhost:/broadcom/ofdpa#

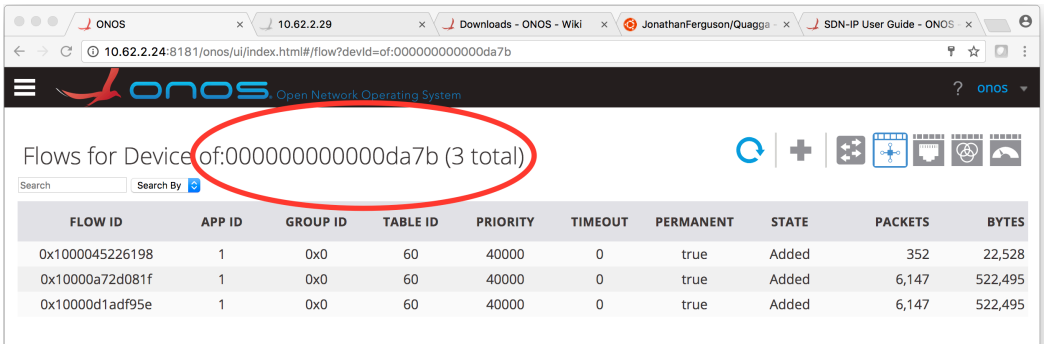
On AG7648-2:

root@localhost:/broadcom/ofdpa# ./client_flowtable_dump
Table ID 60 (ACL Policy): Retrieving all entries. Max entries = 3072, Current entries = 3.
-- inPort:mask = 0:0x0 srcMac:mask = 0000.0000.0000:0000.0000.0000 destMac:mask = 0000.0000.0000:0000.0000.0000 etherType:mask = 0x0806:0xffff | GoTo = 0 (None) outPort = CONTROLLER (Reserved) clearAction | priority = 40000 hard_time = 0 idle_time = 0 cookie = 10
-- inPort:mask = 0:0x0 srcMac:mask = 0000.0000.0000:0000.0000.0000 destMac:mask = 0000.0000.0000:0000.0000.0000 etherType:mask = 0x88cc:0xffff | GoTo = 0 (None) outPort = CONTROLLER (Reserved) clearAction | priority = 40000 hard_time = 0 idle_time = 0 cookie = 12
-- inPort:mask = 0:0x0 srcMac:mask = 0000.0000.0000:0000.0000.0000 destMac:mask = 0000.0000.0000:0000.0000.0000 etherType:mask = 0x8942:0xffff | GoTo = 0 (None) outPort = CONTROLLER (Reserved) clearAction | priority = 40000 hard_time = 0 idle_time = 0 cookie = 7

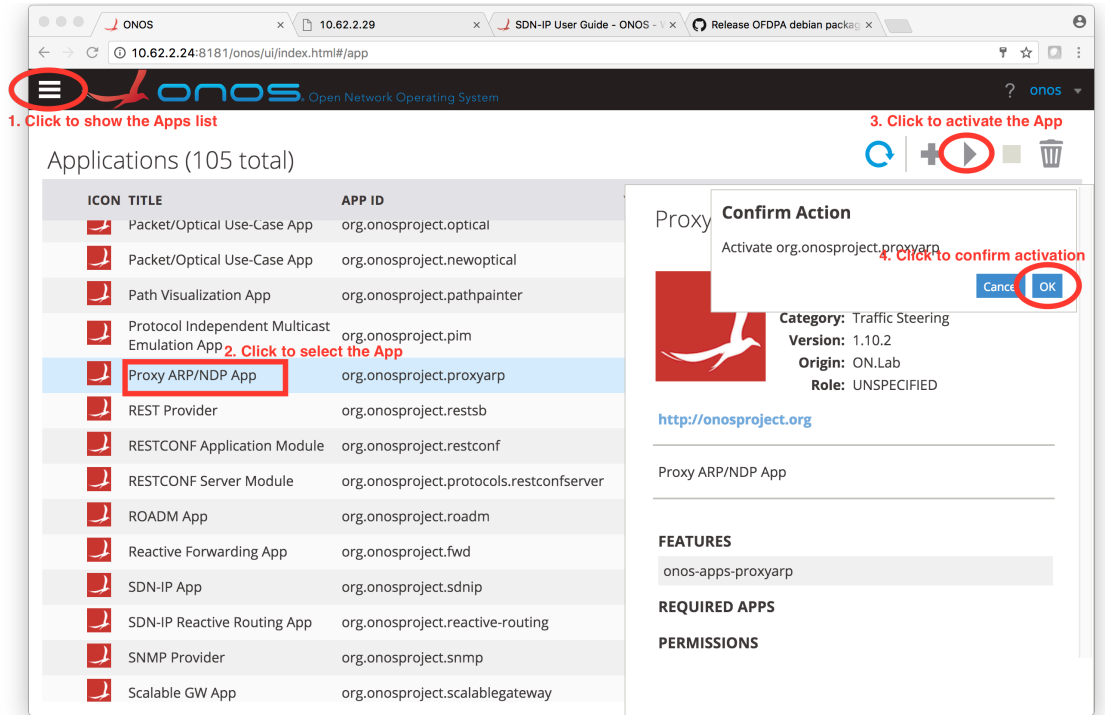
root@localhost:/broadcom/ofdpa#

This are also reflected on the ONOS controllers.





10. Activate Proxy ARP/NDP application on the UI.



a) At this point, RTR1 and RTR2 still shows no learned BGP networks and routes.

On RTR1:

(RTR1) #***show ip route bgp***

Route Codes: C - Connected, S - Static

B - BGP Derived

O - OSPF Derived, IA - OSPF Inter Area

E1 - OSPF External Type 1, E2 - OSPF External Type 2

N1 - OSPF NSSA External Type 1, N2 - OSPF NSSA External Type 2

S U - Unnumbered Peer

L - Leaked Route

K - Kernel, P - Net Prototype

On RTR2:

(RTR2) #***show ip route bgp***

Route Codes: C - Connected, S - Static

B - BGP Derived

O - OSPF Derived, IA - OSPF Inter Area

E1 - OSPF External Type 1, E2 - OSPF External Type 2

N1 - OSPF NSSA External Type 1, N2 - OSPF NSSA External Type 2

S U - Unnumbered Peer

L - Leaked Route

K - Kernel, P - Net Prototype

On BGP Speaker 1:

BGP-SPKR-1# ***sh ip bgp***

No BGP network exists

BGP-SPKR-1#

On BGP Speaker 2:

BGP-SPKR-2# ***sh ip bgp***

No BGP network exists

BGP-SPKR-2#

b) Pings between the legacy routers fail.

(RTR2) #***ping 110.1.1.1***

Pinging 110.1.1.1 with 0 bytes of data:

----110.1.1.1 PING statistics----

3 packets transmitted, 0 packets received, 100% packet loss

round-trip (msec) min/avg/max = <1/<1/<1

(RTR2) #

11. Execute the *sendFlow.sh* script.
    On ONOS-1, execute *'./sendFlow.sh'* in the *'~/opt/onos-ofdpa/switch1'* directory. Note that AG7648-1 elected ONOS-1 as the master. This command MUST be executed on the master ONOS controller for the switch being programmed.

    Similarly, execute *'./sendFlow.sh'* in the *'~/opt/onos-ofdpa/switch2'* directory of ONOS-2. Note that AG7648-2 elected ONOS-2 as the master.

    a) Both switches now have a total of 15 flows each and groups are also added. This is shown via CLI.

    On AG7648-1:

    root@localhost:/broadcom/ofdpa# ***./client_flowtable_dump | grep entries***

    Table ID 60 (ACL Policy): Retrieving all entries. Max entries = 3072, **Current entries = 15.**

    root@localhost:/broadcom/ofdpa# ***./client_grouptable_dump***

    groupId = 0x00010011 (L2 Interface, VLAN ID = 1, Port ID = 17): duration: 739, refCount:4

    bucketIndex = 0: outputPort = 17 (Physical) popVlanTag = 0 allowVlanTranslation = 0

    groupId = 0x00010031 (L2 Interface, VLAN ID = 1, Port ID = 49): duration: 739, refCount:3

    bucketIndex = 0: outputPort = 49 (Physical) popVlanTag = 0 allowVlanTranslation = 0

    groupId = 0x00010032 (L2 Interface, VLAN ID = 1, Port ID = 50): duration: 739, refCount:3

    bucketIndex = 0: outputPort = 50 (Physical) popVlanTag = 0 allowVlanTranslation = 0

    groupId = 0x10000031 (L2 Rewrite, Index = 49): duration: 739, refCount:2

    bucketIndex = 0: referenceGroupId = 0x00010032 vlanId = 0x0000 (VLAN 0) srcMac: 00:00:00:00:00:00 dstMac: 00:18:23:30:8D:6D

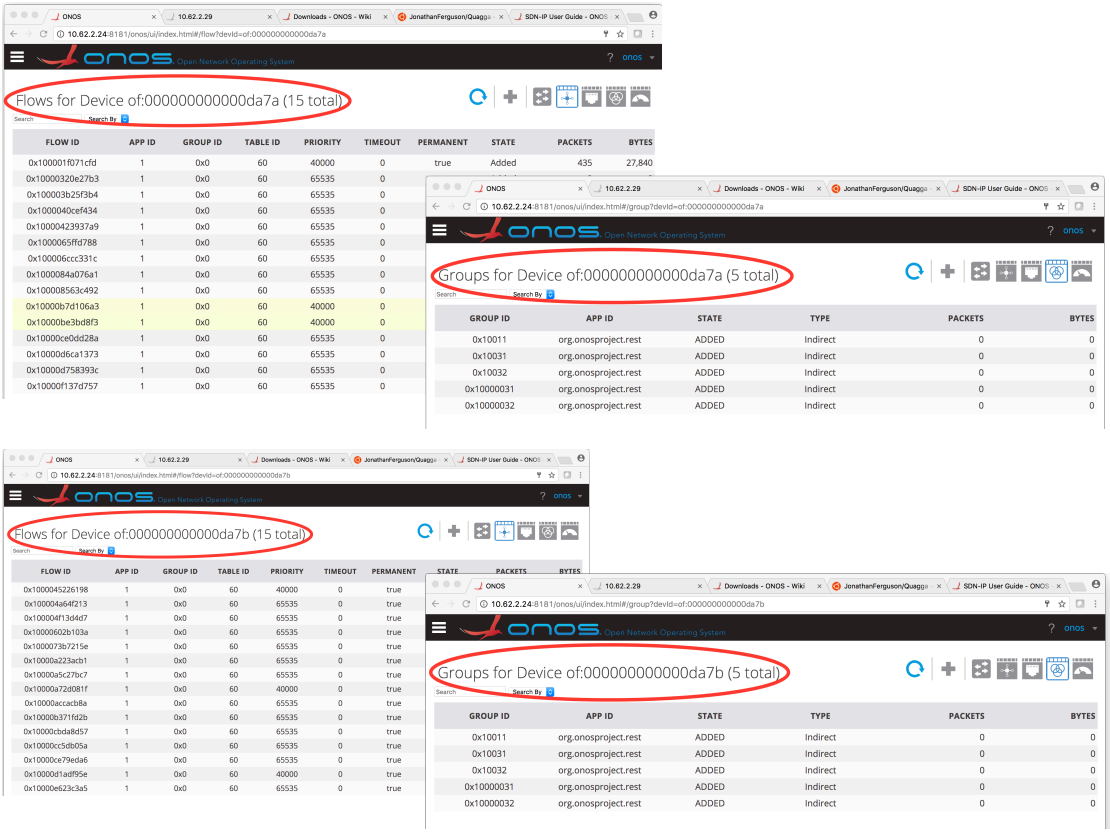
    groupId = 0x10000032 (L2 Rewrite, Index = 50): duration: 739, refCount:2

bucketIndex = 0: referenceGroupId = 0x00010031 vlanId = 0x0000 (VLAN 0)
srcMac: 00:00:00:00:00:00 dstMac: 00:18:23:30:91:37
root@localhost:/broadcom/ofdpa#

On AG7648-2:

root@localhost:/broadcom/ofdpa# ***./client_flowtable_dump | grep entries***
Table ID 60 (ACL Policy): Retrieving all entries. Max entries = 3072, **Current entries = 15**.
root@localhost:/broadcom/ofdpa# ***./client_grouptable_dump***
groupId = 0x00010011 (L2 Interface, VLAN ID = 1, Port ID = 17): duration: 771, refCount:4
bucketIndex = 0: outputPort = 17 (Physical) popVlanTag = 0 allowVlanTranslation = 0
groupId = 0x00010031 (L2 Interface, VLAN ID = 1, Port ID = 49): duration: 771, refCount:3
bucketIndex = 0: outputPort = 49 (Physical) popVlanTag = 0 allowVlanTranslation = 0
groupId = 0x00010032 (L2 Interface, VLAN ID = 1, Port ID = 50): duration: 771, refCount:3
bucketIndex = 0: outputPort = 50 (Physical) popVlanTag = 0 allowVlanTranslation = 0
groupId = 0x10000031 (L2 Rewrite, Index = 49): duration: 771, refCount:2
bucketIndex = 0: referenceGroupId = 0x00010032 vlanId = 0x0000 (VLAN 0)
srcMac: 00:00:00:00:00:00 dstMac: 00:18:23:30:8D:6D
groupId = 0x10000032 (L2 Rewrite, Index = 50): duration: 770, refCount:2
bucketIndex = 0: referenceGroupId = 0x00010031 vlanId = 0x0000 (VLAN 0)
srcMac: 00:00:00:00:00:00 dstMac: 00:18:23:30:91:37
root@localhost:/broadcom/ofdpa#

b) The flows and groups are also displayed on the UI of the controllers.



c) BGP networks and routes are visible on all of the routers.

On RTR1:

(RTR1) #***show ip route bgp***

Route Codes: C - Connected, S - Static

B - BGP Derived

O - OSPF Derived, IA - OSPF Inter Area

E1 - OSPF External Type 1, E2 - OSPF External Type 2

N1 - OSPF NSSA External Type 1, N2 - OSPF NSSA External Type 2

S U - Unnumbered Peer

L - Leaked Route

K - Kernel, P - Net Prototype

B 102.1.1.0/24 [20/0] via 100.1.1.1, 00h:28m:04s, 0/1

B 121.1.1.1/32 [20/0] via 100.1.1.1, 00h:28m:04s, 0/1

On RTR2:

(RTR2) #***show ip route bgp***

Route Codes: C - Connected, S - Static

B - BGP Derived

O - OSPF Derived, IA - OSPF Inter Area

E1 - OSPF External Type 1, E2 - OSPF External Type 2

N1 - OSPF NSSA External Type 1, N2 - OSPF NSSA External Type 2

S U - Unnumbered Peer

L - Leaked Route

K - Kernel, P - Net Prototype

B 100.1.1.0/24 [20/0] via 102.1.1.1, 00h:25m:06s, 0/3

B 110.1.1.1/32 [20/0] via 102.1.1.1, 00h:25m:06s, 0/3

On BGP Speaker 1:

BGP-SPKR-1# ***sh ip bgp***

BGP table version is 0, local router ID is 21.1.1.1

Status codes: s suppressed, d damped, h history, * valid, > best, = multipath,

i internal, r RIB-failure, S Stale, R Removed

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 100.1.1.0/24	100.1.1.11	0		0	100 i
*>i102.1.1.0/24	102.1.1.11	0	100	0	200 i
*> 110.1.1.1/32	100.1.1.11	0		0	100 i
*>i121.1.1.1/32	102.1.1.11	0	100	0	200 i

Total number of prefixes 4

BGP-SPKR-1#

On BGP Speaker 2:

BGP-SPKR-2# ***sh ip bgp***

BGP table version is 0, local router ID is 21.1.1.1

Status codes: s suppressed, d damped, h history, * valid, > best, = multipath,

i internal, r RIB-failure, S Stale, R Removed

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 100.1.1.0/24	100.1.1.11	0		0	100 i
*>i102.1.1.0/24	102.1.1.11	0	100	0	200 i
*> 110.1.1.1/32	100.1.1.11	0		0	100 i
*>i121.1.1.1/32	102.1.1.11	0	100	0	200 i

Total number of prefixes 4

BGP-SPKR-2#

d) Pings between the legacy routers is successful.

RTR1 to RTR2:

(RTR1) #***ping 121.1.1.1***

Pinging 121.1.1.1 with 0 bytes of data:

Reply From 121.1.1.1: icmp_seq = 0. time= 20 msec.

Reply From 121.1.1.1: icmp_seq = 1. time= 36 msec.

Reply From 121.1.1.1: icmp_seq = 2. time= 36 msec.

----121.1.1.1 PING statistics----

3 packets transmitted, 3 packets received, 0% packet loss

round-trip (msec) min/avg/max = 20/30/36

RTR2 to RTR1:

(RTR2) #***ping 110.1.1.1***

Pinging 110.1.1.1 with 0 bytes of data:

Reply From 110.1.1.1: icmp_seq = 0. time= 12 msec.
Reply From 110.1.1.1: icmp_seq = 1. time= 20 msec.
Reply From 110.1.1.1: icmp_seq = 2. time= 20 msec.

----110.1.1.1 PING statistics----
3 packets transmitted, 3 packets received, 0% packet loss
round-trip (msec) min/avg/max = 12/17/20

e) Traceroute shows one hop away.

(RTR2) #***traceroute 110.1.1.1***

Traceroute to 110.1.1.1 ,30 hops max 0 byte packets:

1 110.1.1.1 <1 ms <1 ms <1 ms

Hop Count = 1 Last TTL = 1 Test attempt = 3 Test Success = 3

Findings and Outstanding Issues

- When BGP Router App is activated, the ONOS controller installs BGP flows to the first OFDPA-enabled switch only. Succeeding switches don't get the BGP flows.
- When an ONOS controller is re-started while it's a member of a cluster, the remaining working members experience out-of-memory shortly after.
- When the ONOS controller is left running for some time - say overnight, and then get re-started, the UI will report socket error and a black screen upon login. There are times when not all of the Apps that were active will come back up after starting the ONOS again. However, even after re-activating the apps or even when all of the apps come back active, the same issue persists. The only known solution is to blow up the controller and start all over again.
- ONOS intent-based applications aren't compatible with OFDPA. When the controller pushes these flows (derived from the intents) into the switch, the switch rejects such flows. They will remain in 'Pending Add' state in ONOS.
- We need to make the OFDPA app persistent over a switch re-start.

References

Quagga Install Instructions	https://wiki.ubuntu.com/JonathanFerguson/Quagga
Quagga Images	http://download.savannah.gnu.org/releases/quagga/
ONOS Released Packages	https://wiki.onosproject.org/display/ONOS/Downloads
ONOS-SDN-IP App User Guide	https://wiki.onosproject.org/display/ONOS/SDN-IP+User+Guide
ONOS Tutorials	http://sdnhub.org/tutorials/onos/
Discovery in Software-Defined Networks	http://vlkan.com/blog/post/2013/08/06/sdn-discovery/
ONL Images	https://github.com/DeltaProducts/AG7648
ONL Install Instructions	https://github.com/DeltaProducts/SolutionCenter/blob/master/Install%20ONL.pdf
OFDPA Packages	https://github.com/DeltaProducts/AG7648/releases/tag/1.0
Ubuntu Packages	https://www.ubuntu.com/download
Apache Maven Packages	https://maven.apache.org/download.cgi
Java 1.8 Package	https://java.com/en/download/

Appendix

Script and JSON Files Used in this Test

---

sendFlow.sh

```
#!/bin/sh
set -x
of_controller=10.62.2.24
controller_port=8181
switch_id=of:00:00:00:00:00:00:da:7a
switch_id_url=of%3A000000000000da7a
user_pwd=onos:rocks

#set up L2 interface group
curl -i -v -u $user_pwd -H "Content-Type: application/json"    -d @l2IntGrp_port49.json
http://${of_controller}:$controller_port/onos/v1/groups/$switch_id_url
curl -i -v -u $user_pwd -H "Content-Type: application/json"    -d @l2IntGrp_port50.json
http://${of_controller}:$controller_port/onos/v1/groups/$switch_id_url
curl -i -v -u $user_pwd -H "Content-Type: application/json"    -d @l2IntGrp_port17.json
http://${of_controller}:$controller_port/onos/v1/groups/$switch_id_url

# set up L2 rewrite groups
curl -i -v -u $user_pwd -H "Content-Type: application/json"    -d @l2RwGrp_port49.json
http://$of_controller:$controller_port/onos/v1/groups/$switch_id_url
curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @l2RwGrp_port50.json
http://$of_controller:$controller_port/onos/v1/groups/$switch_id_url

# set up ACL flows
curl -i -v -u $user_pwd -H "Content-Type: application/json"    -d @aclFlow_port492.json
http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core
```

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port493.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port501.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port502.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port1749_icmp.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port1749_tcp.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port1750_icmp.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port1750_tcp.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port4917_icmp.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port5017_tcp.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port5017_icmp.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

curl -i -v -u $user_pwd -H "Content-Type: application/json" -d @aclFlow_port4917_tcp.json http://$of_controller:$controller_port/onos/v1/flows/$switch_id_url?appId=org.onosproject.core

---

aclFlow_port1749_icmp.json

```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
```

```
        "deferred": [
          {
            "type": "GROUP",
            "groupId": "65585"
          }
        ]
    },
    "selector": {
        "criteria": [
          {
            "type": "IN_PORT",
            "port": "17"
          },
          {
              "type": "ETH_TYPE",
              "ethType": "0x0800"
          },
          {
              "type": "IP_PROTO",
              "protocol" : "1"
          },
          {
              "type": "IPV4_DST",
              "ip": "100.1.1.11/32"
          }
        ]
    }
}
```

---

aclFlow_port1749_tcp.json

```
{
```

```
"priority": 65535,
"timeout": 0,
"isPermanent": true,
"deviceId": "of:000000000000da7a",
"tableId": 60,
"treatment": {
   "deferred": [
      {
         "type": "GROUP",
         "groupId": "65585"
      }
   ]
},
"selector": {
   "criteria": [
      {
         "type": "IN_PORT",
         "port": "17"
      },
      {
            "type": "ETH_TYPE",
            "ethType": "0x0800"
      },
      {
            "type": "IP_PROTO",
            "protocol" : "6"
      },
      {
            "type": "IPV4_DST",
            "ip": "100.1.1.11/32"
      }
   ]
```

```
    }
}
```

---

aclFlow_port1750_icmp.json

```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
       "deferred": [
          {
             "type": "GROUP",
             "groupId": "65586"
          }
       ]
    },
    "selector": {
       "criteria": [
          {
             "type": "IN_PORT",
             "port": "17"
          },
          {
                "type": "ETH_TYPE",
                "ethType": "0x0800"
          },
          {
                "type": "IP_PROTO",
                "protocol" : "1"
```

```
            },
            {
                "type": "IPV4_DST",
                "ip": "102.1.1.11/32"
            }
        ]
    }
}
```

---

aclFlow_port1750_tcp.json

```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
        "deferred": [
            {
                "type": "GROUP",
                "groupId": "65586"
            }
        ]
    },
    "selector": {
        "criteria": [
            {
                "type": "IN_PORT",
                "port": "17"
            },
            {
```

```
            "type": "ETH_TYPE",
            "ethType": "0x0800"
        },
        {
            "type": "IP_PROTO",
            "protocol" : "6"
        },
        {
            "type": "IPV4_DST",
            "ip": "102.1.1.11/32"
        }
      ]
    }
}
```

---

aclFlow_port4917_icmp.json

```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
      "deferred": [
        {
          "type": "GROUP",
          "groupId": "65553"
        }
      ]
    },
    "selector": {
```

```
        "criteria": [
          {
            "type": "IN_PORT",
            "port": "49"
          },
          {
              "type": "ETH_TYPE",
              "ethType": "0x0800"
          },
          {
              "type": "IP_PROTO",
              "protocol" : "1"

          }
        ]
    }
}
```

---

aclFlow_port4917_tcp.json

```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
      "deferred": [
        {
          "type": "GROUP",
          "groupId": "65553"
        }
```

```
        ]
    },
    "selector": {
        "criteria": [
            {
                "type": "IN_PORT",
                "port": "49"
            },
            {
                    "type": "ETH_TYPE",
                    "ethType": "0x0800"
            },
            {
                    "type": "IP_PROTO",
                    "protocol" : "6"

            }
        ]
    }
}
```


---

aclFlow_port492.json


```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
        "deferred": [
            {
```

```
            "type": "GROUP",
            "groupId": "268435505"
          }
        ]
    },
    "selector": {
      "criteria": [
        {
          "type": "IN_PORT",
          "port": "49"
        },
        {
            "type": "ETH_TYPE",
            "ethType": "0x0800"
        },
        {
           "type": "IPV4_DST",
           "ip": "121.1.1.0/24"
        }
      ]
    }
}
```


---

aclFlow_port493.json


```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
```

```
        "deferred": [
          {
            "type": "GROUP",
            "groupId": "268435505"
          }
        ]
    },
    "selector": {
        "criteria": [
          {
            "type": "IN_PORT",
            "port": "49"
          },
          {
              "type": "ETH_TYPE",
              "ethType": "0x0800"
          },
          {
              "type": "IPV4_DST",
              "ip": "102.1.1.0/24"
          }
        ]
    }
}
```


---

aclFlow_port5017_icmp.json


```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
```

```
    "tableId": 60,
    "treatment": {
      "deferred": [
        {
          "type": "GROUP",
          "groupId": "65553"
        }
      ]
    },
    "selector": {
      "criteria": [
        {
          "type": "IN_PORT",
          "port": "50"
        },
        {
            "type": "ETH_TYPE",
            "ethType": "0x0800"
        },
        {
            "type": "IP_PROTO",
            "protocol" : "1"

        }
      ]
    }
}
```


---

aclFlow_port5017_tcp.json


```
{
    "priority": 65535,
```

```
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
       "deferred": [
          {
            "type": "GROUP",
            "groupId": "65553"
          }
       ]
    },
    "selector": {
       "criteria": [
          {
            "type": "IN_PORT",
            "port": "50"
          },
          {
                "type": "ETH_TYPE",
                "ethType": "0x0800"
          },
          {
                "type": "IP_PROTO",
                "protocol" : "6"

          }
       ]
    }
}
```


---

aclFlow_port501.json


```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
      "deferred": [
        {
          "type": "GROUP",
          "groupId": "268435506"
        }
      ]
    },
    "selector": {
      "criteria": [
        {
          "type": "IN_PORT",
          "port": "50"
        },
        {
            "type": "ETH_TYPE",
            "ethType": "0x0800"
        },
        {
            "type": "IPV4_DST",
            "ip": "110.1.1.1/32"
        }
      ]
    }
}
```

---

aclFlow_port502.json

```
{
    "priority": 65535,
    "timeout": 0,
    "isPermanent": true,
    "deviceId": "of:000000000000da7a",
    "tableId": 60,
    "treatment": {
       "deferred": [
          {
             "type": "GROUP",
             "groupId": "268435506"
          }
       ]
    },
    "selector": {
       "criteria": [
          {
             "type": "IN_PORT",
             "port": "50"
          },
          {
                "type": "ETH_TYPE",
                "ethType": "0x0800"
          },
          {
                "type": "IPV4_DST",
                "ip": "100.1.1.0/24"
          }
       ]
```

```
    }
}
```

---

l2IntGrp_port17.json

```
{
    "type": "INDIRECT",
    "appCookie": "0x1234abc9",
    "groupId": "65553",
    "buckets": [
      {
        "treatment": {
          "instructions": [
            {
              "type": "OUTPUT",
              "port": "17"
            }
          ]
        }
      }
    ]
}
```

---

l2IntGrp_port49.json

```
{
    "type": "INDIRECT",
    "appCookie": "0x1234abc3",
    "groupId": "65585",
    "buckets": [
      {
        "treatment": {
```

```
            "instructions": [
                {
                   "type": "OUTPUT",
                   "port": "49"
                }
             ]
          }
       }
    ]
}
```

---

l2IntGrp_port50.json

```
{
    "type": "INDIRECT",
    "appCookie": "0x1234abc2",
    "groupId": "65586",
    "buckets": [
     {
        "treatment": {
           "instructions": [
                {
                   "type": "OUTPUT",
                   "port": "50"
                }
             ]
          }
       }
    ]
}
```

---

l2RwGrp_port49.json

```
{
    "type": "INDIRECT",
    "appCookie": "0x1234abc5",
    "groupId": "268435505",
    "buckets": [
     {
       "treatment": {
         "instructions": [
             {
              "type":"L2MODIFICATION",
              "subtype":"ETH_DST",
              "mac":"00:18:23:30:8D:6D"
             },
             {
              "type": "GROUP",
              "groupId": "65586"
            }
          ]
       }
     }
    ]
}
```

---

l2RwGrp_port50.json

```
{
    "type": "INDIRECT",
    "appCookie": "0x1234abc6",
    "groupId": "268435506",
    "buckets": [
     {
```

```
        "treatment": {
          "instructions": [
              {
               "type":"L2MODIFICATION",
               "subtype":"ETH_DST",
               "mac":"00:18:23:30:91:37"
              },
              {
               "type": "GROUP",
               "groupId": "65585"
              }
            ]
          }
        }
    ]
}
```