Chapter 1: Introduction to SDN - Transformation from Legacy to SDN
An Operational View of OpenDaylight

Independent Network Applications

AAA Authorization Filter
OpenDaylight APIs

Control Plane Functions
Embedded Controller Applications
Network Abstractions (Policy/Intent)

Service Abstraction Layer/Core
Messaging (Notifications / RPCs)

Data Plane Elements (Virtual Switches, Physical Device Interfaces)

Standardized Southbound Interfaces & Protocol Plugins

OpenFlow Enabled Devices
Open vSwitch
Additional Virtual & Physical Devices

Application

RESTCONF

Model
Endpoint Registry
Contract Composer
Affinity Service
Inventory
Forwarding Rules Manager

Model
Classic Decomposer
Affinity Decomposer

Model
Native

Traditional Network Elements

Native
Chapter 2: Overview of OpenDaylight
Sample L2Switch Using MD-SAL

1. Add new address
   - L2AddressTracker
   - FlowWriterService
   - TopologyLinkDataChangeHandler

2. Get "external" ports
   - 2.a. Write mac-mac flow when src and dest mac is known.
   - Add flow on entire path (each port) between those hosts
   - 2.b. Get "external" ports

Commit L2-Address

Commit mac-mac flow

On data change (of link)

On packet received

Update flow (SalFlowService Provider)

MD-Sal Data Store
- Configuration
- Operational
  - Flow
  - Topology
  - Inventory
  - L2-Address

MD-Sal
- Two phase Commit
- Packet-Notification

Packet-in

Add flows or teleport packet

OpenFlow plugin

Network

ODL Controlled Network Switch(es)
- S1
- S2
- ... Sn
Chapter 3: OpenDaylight Installation and Deployment

### Downloads

<table>
<thead>
<tr>
<th>Release</th>
<th>Edition</th>
<th>Version</th>
<th>Release date</th>
<th>Downloads</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boron-SR1</td>
<td>n/a</td>
<td>n/a</td>
<td>November 3, 2016</td>
<td>Pre-Built Tar</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Pre-Built Zip</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Virtual Tenant Network (VTN) Coordinator</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NeXt UI Toolkit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OpFlex</td>
</tr>
</tbody>
</table>

```
learningod7@ODL01:~$ cat /etc/issue
Ubuntu 16.04.1 LTS

learningod7@ODL01:~$ ```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto ens33
iface ens33 inet static
  address 192.168.20.51
  netmask 255.255.255.0
  gateway 192.168.20.2
  dns-nameservers 4.2.2.4 8.8.8.8

$ java -version
openjdk version "1.8.0_111"
OpenJDK Runtime Environment (build 1.8.0_111-b14-2ubuntu0.16.04.2-b14)
OpenJDK 64-Bit Server VM (build 25.111-b14, mixed mode)

$ sudo update-alternatives --config java
There is only one alternative in link group java (providing /usr/bin/java): /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java
Nothing to configure.

PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/bin/jvm:
JAVA_HOME="/usr/lib/jvm/java-8-openjdk-amd64/jre"
Hyper-V feature is not enabled.
Do you want to enable it for Docker to be able to work properly?
Your computer will restart automatically.
Note: Docker Toolbox will no longer work.

```
PS C:\Users\Reza> docker version
Client:
  Version: 17.03.0-CE
  API version: 1.26
  Go version: go1.7.5
  Git commit: 6d6d822
  Built: Thu Feb 23 10:40:39 2017
  OS/Arch: windows/amd64

Server:
  Version: 17.03.0-CE
  API version: 1.26 (minimum version 1.12)
  Go version: go1.7.5
  Git commit: 3ae2ad73
  Built: Tue Feb 28 07:52:04 2017
  OS/Arch: linux/amd64
  Experimental: true
```

```
PS C:\Users\Reza> docker pull learningopendaylight/boron
Using default tag: latest
latest: Pulling from learningopendaylight/boron
Digest: sha256:6704534b5a5c4f23b023bf0796037f02a6f663140a222e7c5a66a46455ed9e62cfd3bb
Status: Downloaded newer image for learningopendaylight/boron:latest
```

```
PS C:\Users\Reza> docker images
REPOSITORY               TAG       IMAGE ID             CREATED              SIZE
learningopendaylight/boron latest 08a356af9d86 3 hours ago 1.88GB
```

PS C:\Users\Reza> docker run -it -p 8181:8181 learningopendaylight/boron
karaf: JAVA_HOME not set; results may vary
Apache Karaf starting up. Press Enter to open the shell now...
100% [==================================]
Karaf started in 3s. Bundle stats: 64 active, 64 total

OpenDaylight

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown OpenDaylight.

opendaylight-user@root>

opendaylight-user@root>feature:install odl-dlux-all
opendaylight-user@root>

Please Sign In

OPEN DAYLIGHT

Username
Password

Remember Me
Login

PS C:\Users\Reza> shutdown
Confirm: halt instance root (yes/no): yes
PS C:\Users\Reza>
<table>
<thead>
<tr>
<th>Line</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FROM ubuntu:16.04</td>
</tr>
<tr>
<td>2</td>
<td>RUN apt-get -y update &amp;&amp; apt-get install -y openjdk-8-jre wget</td>
</tr>
<tr>
<td>3</td>
<td>RUN mkdir /opt/odl</td>
</tr>
<tr>
<td>4</td>
<td>RUN wget -O /opt/odl/odl.tar.gz <a href="https://nexus.opendaylight.org/content/repositories.opendaylight.release/org/opendaylight/integration/distribution-karaf/0.5.2-Boron-SR2/distribution-karaf-0.5.2-Boron-SR2.tar.gz">https://nexus.opendaylight.org/content/repositories.opendaylight.release/org/opendaylight/integration/distribution-karaf/0.5.2-Boron-SR2/distribution-karaf-0.5.2-Boron-SR2.tar.gz</a></td>
</tr>
<tr>
<td>5</td>
<td>RUN tar -xzf /opt/odl/odl.tar.gz -C /opt/odl/</td>
</tr>
<tr>
<td>6</td>
<td>RUN mv /opt/odl/dist* /opt/odl/boron</td>
</tr>
<tr>
<td>7</td>
<td>CMD /opt/odl/boron/bin/karaf</td>
</tr>
</tbody>
</table>
Chapter 4: Building a Virtual SDN Test Lab with Virtual Switches
mininet@mininet-vm:/$ sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>

mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo: s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet>  
```
mininet> h1 ifconfig
h1-eth0  Link encap:Ethernet  HWaddr 58:44:58:97:74
        inet addr:10.0.0.1  Bcast:10.255.255.255  Mask:255.0.0.0
        UP  BROADCAST  RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:0  errors:0  dropped:0  overruns:0  frame:0
        TX packets:0  errors:0  dropped:0  overruns:0  carrier:0
        collisions:0  txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo    Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        UP  LOOPBACK  RUNNING  MTU:65536  Metric:1
        RX packets:0  errors:0  dropped:0  overruns:0  frame:0
        TX packets:0  errors:0  dropped:0  overruns:0  carrier:0
        collisions:0  txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

mininet> h2 ifconfig
h2-eth0  Link encap:Ethernet  HWaddr 5e:80:93:72:8b:d2
        inet addr:10.0.0.2  Bcast:10.255.255.255  Mask:255.0.0.0
        UP  BROADCAST  RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:0  errors:0  dropped:0  overruns:0  frame:0
        TX packets:0  errors:0  dropped:0  overruns:0  carrier:0
        collisions:0  txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo    Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        UP  LOOPBACK  RUNNING  MTU:65536  Metric:1
        RX packets:0  errors:0  dropped:0  overruns:0  frame:0
        TX packets:0  errors:0  dropped:0  overruns:0  carrier:0
        collisions:0  txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

mininet> h2 cat /etc/issue
Ubuntu 14.04 LTS

mininet>
```

```
mininet> h1 bash
root@mininet-vm:~#
root@mininet-vm:~# cd /bin
root@mininet-vm:~# ls
bin  dev  home  lib  lost+found  mnt  proc  run  srv  tmp  var
root@mininet-vm:~#
```
Download/Get Started With Mininet

The easiest way to get started is to download a pre-packaged Mininet:Ubuntu VM. This VM includes Mininet itself, all OpenFlow binaries and tools pre-installed, and tweaks to the kernel configuration to support larger Mininet networks.

- Option 1: Mininet VM Installation (easy, recommended)
- Option 2: Native Installation from Source
- Option 3: Installation from Packages
- Option 4: Upgrading an existing Mininet Installation

Option 1: Mininet VM Installation (easy, recommended)

VM installation is the easiest and most foolproof way of installing Mininet, so it’s what we recommend to start with.

Follow these steps for a VM install:
1. Download the Mininet VM image.
2. Download and install a virtualization system. We recommend VirtualBox (free, GPL) because it is free and works on OS X, Windows, and Linux (though it’s slightly slower than VMware in our tests). You can also use Qemu for any platform, VMware Workstation for Windows or Linux, VMware Fusion for Mac, or KVM (free, GPL) for Linux.
3. Sign up for the mininet-discuss mailing list. This is the source for Mininet support and discussion with the friendly Mininet community. :-)
4. Run through the VM Setup Notes to log in to the VM and customize it as desired.
5. Follow the Walkthrough to get familiar with Mininet commands and typical usage.
The selected virtual machine is inaccessible. Please inspect the error message shown below and press the Refresh button if necessary.

Import Virtual Appliance

Appliance to import

VirtualBox currently supports importing appliances saved in the Open Virtualization Format (OVF). To continue, select the file to import below.
Appliance settings

These are the virtual machines contained in the appliance and the suggested settings of the imported VirtualBox machines. You can change many of the properties shown by double-clicking on the items and disable others using the check boxes below.

<table>
<thead>
<tr>
<th>Description</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Virtual System 1</strong></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Mininet-VM</td>
</tr>
<tr>
<td>Guest OS Type</td>
<td>Ubuntu (64-bit)</td>
</tr>
<tr>
<td>CPU</td>
<td>1</td>
</tr>
<tr>
<td>RAM</td>
<td>1024 MB</td>
</tr>
<tr>
<td>USB Controller</td>
<td></td>
</tr>
<tr>
<td>Network Adapter</td>
<td>Intel PRO/1000 MT Server (82545E)</td>
</tr>
<tr>
<td>Storage Controller (SCSI)</td>
<td>LsiLogic</td>
</tr>
<tr>
<td>Virtual Disk Image</td>
<td>C:\Users\Reza\VirtualBox VMs\Mininet...</td>
</tr>
</tbody>
</table>

- Reinitialize the MAC address of all network cards
- Appliance is not signed

[Buttons: Restore Defaults, Import, Cancel]
Import Virtual Machine

Store the new Virtual Machine
Provide a name and local storage path for the new virtual machine.

Name for the new virtual machine:
MinINET

Storage path for the new virtual machine:
C:\VIRTUAL-MACHINES\MinINET

[Import] [Cancel]
mininet@mininet-vm:/$/ sudo mn --controller=remote,ip=192.168.20.51 --topo=linear,2 --mac
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2
*** Adding links:
(h1, s1) (h2, s2) (s2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...
*** Starting CLI:
mininet>
Wireshark - Packet 78, wireshark_2880CB8F-F843-4526-953D-49773024E253_20170107163529_a64456

- Frame 78: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
  - Ethernet II, Src: VMware_69:36:bc (00:0c:29:69:36:bc), Dst: VMware_44:f0:0c (00:0c:29:44:f0:0c)
  - Transmission Control Protocol, Src Port: 8833, Dst Port: 87568, Seq: 1, Ack: 1, Len: 8
  - OpenFlow 1.0
      .000 0001 - Version: 1.0 (0x01)
      Type: OFPT_HELLO (0)
      Length: 8
      Transaction ID: 62

Wireshark - Packet 84, wireshark_2880CB8F-F843-4526-953D-49773024E253_20170107163529_a64456

- Frame 84: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
  - Ethernet II, Src: VMware_44:f0:0c (00:0c:29:44:f0:0c), Dst: VMware_69:36:bc (00:0c:29:69:36:bc)
  - Transmission Control Protocol, Src Port: 6633, Dst Port: Port 37569, Seq: 1, Ack: 9, Len: 8
  - OpenFlow 1.0
      .000 0001 - Version: 1.0 (0x01)
      Type: OFPT_HELLO (0)
      Length: 8
      Transaction ID: 65

mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2)  56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.280 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.168 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.178 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.197 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.159 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.161 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.148 ms

--- 10.0.0.2 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 600ms
round-trip min/avg/max/mdev = 0.148/0.184/0.280/0.043 ms
mininet>
Ethernet II, Src: Vmware_69:36:bc (00:0c:29:69:36:bc), Dst: Vmware_44:f0:0c (00:0c:29:44:f0:0c)
OpenFlow 1.0
  .000 0001 = Version: 1.0 (0x01)
  Type: OFPT_PACKET_IN (10)
  Length: 60
  Transaction ID: 0
  Buffer Id: 0xffffffff
  Total length: 42
  in port: 1
  Reason: Action explicitly output to controller (1)
  Pad: 0

Ethernet II, Src: 00:00:00:00:00:01 (00:00:00:00:00:01), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPV4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: 00:00:00:00:00:01 (00:00:00:00:00:01)
  Sender IP address: 16.0.0.1
  Target MAC address: 00:00:00:00:00:00 (00:00:00:00:00:00)
  Target IP address: 16.0.0.2

245 7.031936 192.168.20.51 192.168.20.55 OpenFlow 74 Type: OFPT_BARRIER_REQUEST
246 7.032310 192.168.20.51 192.168.20.55 OpenFlow 74 Type: OFPT_BARRIER_REPLY
251 7.521639 192.168.20.51 192.168.20.55 OpenFlow 146 Type: OFPT_FLOW_MOD
252 7.521659 192.168.20.51 192.168.20.55 OpenFlow 154 Type: OFPT_FLOW_MOD
254 7.522124 192.168.20.51 192.168.20.55 OpenFlow 154 Type: OFPT_FLOW_MOD
255 7.522342 192.168.20.51 192.168.20.55 OpenFlow 146 Type: OFPT_FLOW_MOD
257 7.552086 192.168.20.51 192.168.20.55 OpenFlow 74 Type: OFPT_BARRIER_REQUEST
258 7.552302 192.168.20.51 192.168.20.55 OpenFlow 74 Type: OFPT_BARRIER_REPLY

[Diagram of network topology with host and openflow nodes]
Ethernet II, Src: VMware_44:f0:0c (00:0c:29:44:f0:0c), Dst: VMware_69:36:bc (00:0c:29:69:36:bc)

OpenFlow 1.0

0x00 0001 = Version: 1.0 (0x01)
  Type: OFPT_FLOW_MOD (14)
  Length: 80
  Transaction ID: 20
  Wildcards: 3678462
  In port: 2
  Ethernet source address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Ethernet destination address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Input VLAN id: 65535
  Input VLAN priority: 0
  Pad: 0
  DL type: 0
  IP ToS: 0
  IP protocol: 0
  Pad: 0
  Source Address: 0.0.0.0
  Destination Address: 0.0.0.0
  Source Port: 0
  Destination Port: 0
  Cookie: 0xb000000000000010
  Command: New flow (0)
  Idle time-out: 0
  Hard time-out: 0
  Priority: 2
  Buffer Id: 0xffffffff
  Out port: 65535
  Flags: 0
<table>
<thead>
<tr>
<th>Protocol</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet II</td>
<td>VMware_44:f0:0c (00:0c:29:44:f0:0c)</td>
<td>VMware_69:36:bc (00:0c:29:69:36:bc)</td>
</tr>
<tr>
<td>Internet Protocol Version 4</td>
<td>192.168.20.51</td>
<td>192.168.20.55</td>
</tr>
</tbody>
</table>

**OpenFlow 1.0**

- Version: 1.0 (0x01)
- Type: OFPT_FLOW_MOD (14)
- Length: 88
- Transaction ID: 21
- Wildcards: 3678462

**In port: 1**

- Ethernet source address: 00:00:00:00:00:00
- Ethernet destination address: 00:00:00:00:00:00
- Input VLAN id: 65535
- Input VLAN priority: 0
- Pad: 0
- Dl type: 0
- IP ToS: 0
- IP protocol: 0
- Pad: 0000
- Source Address: 0.0.0.0
- Destination Address: 0.0.0.0
- Source Port: 0
- Destination Port: 0
- Cookie: 0x2d00000000000000
- Command: New flow (0)
- Idle time-out: 0
- hard time-out: 0
- Priority: 2
- Buffer Id: 0xffffffff

**Out port: 65535**

Flags: 0
OpenFlow 1.0

Version: 1.0 (0x01)
Type: OFPT_FLOW_MOD (14)
Length: 88
Transaction ID: 20
Wildcards: 3678462

In port: 1
Ethernet source address: 00:00:00:00:00:00 (00:00:00:00:00:00)
Ethernet destination address: 00:00:00:00:00:00 (00:00:00:00:00:00)
Input VLAN id: 65535
Input VLAN priority: 0
Pad: 00
D1 type: 0
IP ToS: 0
IP protocol: 0
Pad: 0000
Source Address: 0.0.0.0
Destination Address: 0.0.0.0
Source Port: 0
Destination Port: 0
Cookie: 0x2b00000000000012
Command: New flow (0)
Idle time-out: 0
Hard time-out: 0
Priority: 2
Buffer Id: 0xffffffff
Out port: 65535
Flags: 0
Ethernet II, Src: Vmware_44:f0:0c (00:0c:29:44:f0:0c), Dst: Vmware_69:36:bc (00:0c:29:69:36:bc)
OpenFlow 1.0

Version: 1.0 (0x01)
Type: OFPT_FLOW_MOD (14)
Length: 80
Transaction ID: 21
Wildcards: 3678462
In port: 2
Ethernet source address: 00:00:00:00:00:00 (00:00:00:00:00:00)
Ethernet destination address: 00:00:00:00:00:00 (00:00:00:00:00:00)
Input VLAN id: 65535
Input VLAN priority: 0
Pad: 0
D1 type: 0
IP ToS: 0
IP protocol: 0
Pad: 0000
Source Address: 0.0.0.0
Destination Address: 0.0.0.0
Source Port: 0
Destination Port: 0
Cookie: 0x2b000000000000013
Command: New flow (0)
Idle time-out: 0
Hard time-out: 0
Priority: 2
Buffer Id: 0xffffffff
Out port: 65535
Flags: 0

---

**Diagram:**

- **H1** (10.0.0.1/24)
- **S1** (Port 1: openflow:1, Port 2: openflow:1)
- **S2** (Port 1: openflow:2)
- **H2** (10.0.0.2/24)

**Network Configuration:**

- Host: 00:00:00:00:00:01
- Host: 00:00:00:00:00:02
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s2-eth1
s1 lo: s1-eth1:h1-eth0 s1-eth2:s2-eth2
s2 lo: s2-eth1:h2-eth0 s2-eth2:s1-eth2
c0
mininet>

mininet> links
h1-eth0<-->s1-eth1 (OK OK)
h2-eth0<-->s2-eth1 (OK OK)
s2-eth2<-->s1-eth2 (OK OK)
mininet>

mininet> nodes
available nodes are:
c0 h1 h2 s1 s2
mininet>

mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=19436>
<Host h2: h2-eth0:10.0.0.2 pid=19439>
<OVS Switch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=19444>
<OVS Switch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None pid=19447>
<RemoteController{'ip': '192.168.20.51'} c0: 192.168.20.51:6633 pid=19430>
mininet>

mininet> h1 ifconfig
*** errRun: ['stty', '-icanon', 'min', '1']
Oh1-eth0  Link enca:Ethernet  Hwa:00:00:00:00:00:01
  inet addr:10.0.0.1 Bcast:10.255.255.255 Mask:255.0.0.0
  UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
lo  Link enca:Local Loopback
  inet addr:127.0.0.1 Mask:255.0.0.0
  UP LOOPBACK RUNNING MTU:65536 Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:0
  RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
mininet>
mininet> h1 ping h2
*** errRun: ['stty', '-icanon', 'min', '1']
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.169 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.155 ms
^C
--- 10.0.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.155/0.162/0.169/0.007 ms

mininet> pingall
*** Ping: testing ping reachability
h1 -> *** h1 : ('ping -c1 10.0.0.2',)
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.268 ms
--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.268/0.268/0.268/0.000 ms

h2 -> *** h2 : ('ping -c1 10.0.0.1',)
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.144 ms
--- 10.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.144/0.144/0.144/0.000 ms

h1
*** Results: 0% dropped (2/2 received)

mininet> pingpair
h1 -> *** h1 : ('ping -c1 10.0.0.2',)
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.383 ms
--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.383/0.383/0.383/0.000 ms

h2 -> *** h2 : ('ping -c1 10.0.0.1',)
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.127 ms
--- 10.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.127/0.127/0.127/0.000 ms

h1
*** Results: 0% dropped (2/2 received)

mininet>
mininet> h1 python -m SimpleHTTPServer 80 &
mininet> h2 wget -O - h1
2016-12-19 19:23:46 -- http://10.0.0.1/
Connecting to 10.0.0.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 802 [text/html]
Saving to: `STDOUT`

0% [ ] 0 --.-K/s

<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href=".bash_history">.bash_history</a></li>
<li><a href=".bash_logout">.bash_logout</a></li>
<li><a href=".bashrc">.bashrc</a></li>
<li><a href=".cache">.cache</a></li>
<li><a href=".gitconfig">.gitconfig</a></li>
<li><a href=".mininet_history">.mininet_history</a></li>
<li><a href=".profile">.profile</a></li>
<li><a href=".rnd">.rnd</a></li>
<li><a href=".wireshark">.wireshark</a></li>
<li><a href="install-mininet-vm.sh">install-mininet-vm.sh</a></li>
<li><a href="loxigen">loxigen</a></li>
<li><a href="mininet">mininet</a></li>
<li><a href="oflops">oflops</a></li>
<li><a href="oftest">oftest</a></li>
<li><a href="openflow">openflow</a></li>
<li><a href="pox">pox</a></li>
</ul>
<hr>
</body>
</html>

100%[==================================] 802 --.-K/s in 0s

2016-12-19 19:23:46 (171 MB/s) - written to stdout [802/802]

mininet>
mininet> dpctl dump-flows

*** s1 (**ovs-ofctl**: 'dump-flows', **ovsSwitch s1**: lo:127.0.0.1, s1-eth1:None, s1-eth2:None pid=20685)

**FLOW** reply (stdout):
  cookie=0x2000000000000000, duration=927.436s, table=0, n_packets=4, n_bytes=280, idle_age=920, priority=2, in_port=1, actions=output:7, CONTROLLER
  offset=0

**FLOW** reply (stdout):
  cookie=0x2000000000000000, duration=927.436s, table=0, n_packets=4, n_bytes=280, idle_age=920, priority=2, in_port=2, actions=output:12, CONTROLLER
  offset=0

**FLOW** reply (stdout):
  cookie=0x2000000000000000, duration=930.441s, table=0, n_packets=0, n_bytes=0, idle_age=930, priority=0, actions=drop

**FLOW** reply (stdout):
  cookie=0x2000000000000000, duration=930.441s, table=0, n_packets=0, n_bytes=0, idle_age=930, priority=0, actions=drop

*** s2 (**ovs-ofctl**: 'dump-flows', **ovsSwitch s2**: lo:127.0.0.1, s2-eth1:None, s2-eth2:None pid=20688)

**FLOW** reply (stdout):
  cookie=0x2000000000000000, duration=927.436s, table=0, n_packets=4, n_bytes=280, idle_age=920, priority=2, in_port=1, actions=output:12, CONTROLLER
  offset=0

**FLOW** reply (stdout):
  cookie=0x2000000000000000, duration=927.436s, table=0, n_packets=4, n_bytes=280, idle_age=920, priority=2, in_port=2, actions=output:12, CONTROLLER
  offset=0

**FLOW** reply (stdout):
  cookie=0x2000000000000000, duration=933.765s, table=0, n_packets=0, n_bytes=0, idle_age=933, priority=0, actions=drop

**FLOW** reply (stdout):
  cookie=0x2000000000000000, duration=933.765s, table=0, n_packets=0, n_bytes=0, idle_age=933, priority=0, actions=drop

```python

```
# Chapter 5: Basic Networking with OpenDaylight

![Topology Diagram]

```bash
learningodl@ODL01:~$ netstat -ln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address     Foreign Address        State
tcp   0      0      0.0.0.0:22       0.0.0.0:*           LISTEN
tcp   0      0      0::8080          :::*                LISTEN
tcp   0      0      0::8161          :::*                LISTEN
tcp   0      0      0::22            :::*                LISTEN
tcp   0      0      127.0.0.1:2550   :::*                LISTEN
tcp   0      0      0::8185          :::*                LISTEN
tcp   0      0      0::44378         :::*                LISTEN
tcp   0      0      0::4444          :::*                LISTEN
tcp   0      0      0::6653          :::*                LISTEN
tcp   0      0      0::42719         :::*                LISTEN
tcp   0      0      127.0.0.1:42655  :::*                LISTEN
tcp   0      0      0::8101          :::*                LISTEN
tcp   0      0      0::6633          :::*                LISTEN
Active UNIX domain sockets (only servers)
Proto RefCnt Flags  Type      State       I-Node Path
unix  2       ACC STREAM LISTENING 14323 /run/user/1001/systemd/private
unix  2       ACC SEQPACKET LISTENING 11180 /run/udev/control
unix  2       ACC STREAM LISTENING 11174 /run/systemd/private
unix  2       ACC STREAM LISTENING 11191 /run/systemd/journal/stdout
unix  2       ACC STREAM LISTENING 11204 /run/systemd/fsck.progress
unix  2       ACC STREAM LISTENING 13766 /run/uuid/uuid
unix  2       ACC STREAM LISTENING 16563 /var/run/dbus/system_bus_socket
unix  2       ACC STREAM LISTENING 13767 /var/lib/1xd/unix.socket
```

---

### OpenDaylight SDN Controller

![OpenDaylight Controller]
Ethernet II, Src: VMware_44:f0:8c (00:0c:29:44:f0:8c), Dst: VMware_69:36:bc (00:0c:29:69:36:bc)

OpenFlow 1.0

Type: OFPT_FLOW_MOD (16)

Ethernet source address: 00:00:00:00:00:00 (00:00:00:00:00:00)
Ethernet destination address: 00:00:00:00:00:00 (00:00:00:00:00:00)
Input VLAN id: 0
Input VLAN priority: 0
Pad: 0
Dl type: 0
IP Tos: 0
IP protocol: 0
Pad: 0
Source Address: 0.0.0.0
Destination Address: 0.0.0.0
Source Port: 0
Destination Port: 0
Cookie: 0x2e0000000000000c8
Command: New flow (0)
Idle time-out: 1000
hard time-out: 3000
Priority: 10
Buffer Id: 0xffffffff
Out port: 85535
Flags: 0
Host 1 can communicate with Host 3 but not with Host 2 and Host 4

VLAN | Member Ports
---|-----------------
100 | Port 1,3
200 | Port 2,4
<table>
<thead>
<tr>
<th>Name of element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual node</td>
<td></td>
</tr>
<tr>
<td>vBridge</td>
<td>Logical representation of an L2 switch function</td>
</tr>
<tr>
<td>vRouter</td>
<td>Logical representation of a router function</td>
</tr>
<tr>
<td>vTep</td>
<td>Logical representation of Tunnel End Point (TEP)</td>
</tr>
<tr>
<td>vTunnel</td>
<td>Logical representation of Tunnel</td>
</tr>
<tr>
<td>vBypass</td>
<td>Logical representation of the connectivity between controlled networks</td>
</tr>
<tr>
<td>Virtual interface</td>
<td>interface</td>
</tr>
<tr>
<td>Virtual link</td>
<td>vLink</td>
</tr>
</tbody>
</table>
Chapter 6: Overview of OpenDaylight Applications
mvn archetype:generate 
-DarchetypeGroupId=org.opendaylight.controller 
-DarchetypeVersion=1.2.0-Boron 
-DarchetypeCatalog=https://nexus.opendaylight.org/content/repositories/public/

INFO Scanning for projects...
INFO Building Maven Stub Project (No POM)...
INFO Generating project in Interactive mode
INFO No catalog defined. Using internal catalog
WARNING Archetype not found in any catalog. Falling back to central repository (http://repo.maven.apache.org)
WARNING] Use -DarchetypeRepository=<your repository> if archetype's repository is elsewhere.
Define value for property 'groupId': org.opendaylight.example
Define value for property 'artifactId': example
INFO Using property: version = 0.1.0-SNAPSHOT
Define value for property 'package' org.opendaylight.example:
Define value for property 'classPrefix' Example: : ${artifactId.substring(0,1).toUpperCase()}${artifactId.substring(1)}
INFO Using property: copyrightYear = 2016
Confirm properties configuration:
groupId: org.opendaylight.example
artifactId: example
version: 0.1.0-SNAPSHOT
package: org.opendaylight.example
classPrefix: ${artifactId.substring(0,1).toUpperCase()}${artifactId.substring(1)}
copyright: learningodl
copyrightYear: 2016
Y: 
Using following parameters for creating project from Archetype: openDayli
Parameter: groupId, Value: org.opendaylight.example
Parameter: artifactId, Value: example
Parameter: version, Value: 0.1.0-SNAPSHOT
Parameter: package, Value: org.opendaylight.example
Parameter: packageInPathFormat, Value: org.opendaylight/example
Parameter: classPrefix, Value: Example
Parameter: package, Value: org.opendaylight.example
Parameter: version, Value: 0.1.0-SNAPSHOT
Parameter: copyright, Value: learningodl
Parameter: groupId, Value: org.opendaylight.example
Parameter: artifactId, Value: example
Parameter: copyrightYear, Value: 2016
WARNING] Don't override file /home/learningodl/example/pom.xml
INFO Project created from Archetype in dir: /home/learningodl/example
BUILD SUCCESS
INFO Total time: 31.502 s
INFO Finished at: 2017-05-18T14:01:17-07:00
INFO Final Memory: 16M/167M
Learningodl@ODL01:~$ cd example
Learningodl@ODL01:~/example$ ls
api artifacts cli deploy-site.xml features impl it karaf pom.xml src

| INFO | Resources/default-site-macros.vm |
| INFO | --- maven-site-plugin:3.5.1:attach-descriptor (generate-site) @ example-aggregator --- |
| INFO | Reactor Summary: |
| INFO | example-api SUCCESS [01:22 min] |
| INFO | example-impl SUCCESS [36.386 s] |
| INFO | example-cl1 SUCCESS [36.794 s] |
| INFO | example-features SUCCESS [07:29 min] |
| INFO | example-karaf SUCCESS [01:39 min] |
| INFO | example-artifacts SUCCESS [3.384 s] |
| INFO | example-it SUCCESS [03:23 min] |
| INFO | example SUCCESS [01:13 min] |
| INFO | BUILD SUCCESS |
| INFO | Total time: 16:35 min |
| INFO | Finished at: 2017-05-18T14:21:43-07:00 |
| INFO | Final Memory: 226M/721M |

Learningodl@ODL01:~/example$ ./karaf
Apache Karaf starting up. Press Enter to open the shell now...
100% [================================================================]  

Karaf started in 51s. Bundle stats: 284 active, 284 total

OpenDaylight

Hit 'ctab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit 'ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown OpenDaylight.

opendaylight-user@root> log:display | grep Example
2017-05-18 14:23:43 640 | INFO | rint Extender: 2 | ExampleProvider | ExampleProvider Session Initiated
OpenDayLight

Hit <tab> for a list of available commands and <[cmd] > help for help on a specific command. Hit <ctrl-d> or type 'system:shutdown' or 'logout' to shutdown OpenDayLight.

2017-04-21 15:16:49,435 | INFO | 169 - org.opendaylight.example.impl - 0.1.0-SNAPSHOT | ExampleProvider | Example Provider Session Initiated
Chapter 7: Building SDN Applications for OpenDaylight
Chapter 8: Network Function Virtualization
NFV

Multiple Virtual Machines
Virtual Firewall
Virtual Router
Virtual Load Balancer
Hypervisor

Multiple Virtual Machines
Virtual Firewall
Virtual Router
Virtual Load Balancer
Hypervisor

High Availability
HA

Commodity X86
Server HW
Copper & Fiber
interfaces
Sending Http Request towards the Virtual IP on Load balancer

User → DMZ Virtual Firewall → Virtual Loadbalancer → NFV → Web Server

Return Traffic from Server back to User
OpenFlow Enabled Switch

User PC
10.1.1.10/24
GW: 10.1.1.1

User accessing crm.localweb.com
Resolving to VIP of 192.168.55.55

Firewall
Minimum 2 zones of Trust and UnTrust

Load balancer
VIP for CRM: 192.168.55.55

Firewall has a routing statement to route
192.168.55.0/24 to Load Balancer

Web Server
172.16.0.10/24
GW: 172.16.0.1

Real Server LAN
172.16.0.4/24

Real Server LAN
172.16.0.5/24

Trust Zone
192.168.11.24

VIP LAN
192.168.11.24
SDN Controller (OpenDayLight) defines what traffic should go to which tools.

For example:
1. Traffic from VM1, hosted on VMware node should go to FireEye.
2. TCP traffic from VM hosted in OpenStack should go to Wireshark tool #2.
Chapter 9: Building a Software-Driven Data Center with OpenDaylight
VTEP (VXLAN Tunnel EndPoint) is the VXLAN encapsulation engine.

Layer 3 ECMP Load balancing.