Software Defined Networking (SDN)
Lab 1: OpenFlow Communications

ACI-REF

Advanced CyberInfrastructure –
Research and Education Facilitators

Version 1.1

ACI-REF: OpenFlow Lab Guide

Purpose:

The purpose of this lab is to introduce the operational concepts of OpenFlow communications in a working simulation. The user will become familiar with emulation tools, virtual networking environments, as well as software defined network controllers and their communications with network devices.

Prerequisite Knowledge:

The user should have a basic knowledge of:

- Operating System (OS) Graphical User Interface (GUI) navigation.
- Comfort with typing text into a command or shell window.
- Basic grasp of computer networking terms such as "host", "switch", and "IP Address".

Lab Requirements:

- A laptop, mobile device, or access to a remote device that meets the basic hardware requirements for running Oracle's Virtual Box.

- A functioning installation of the latest version of Oracle's Virtual Box, available via: https://www.virtualbox.org/wiki/Downloads. Depending on your host device, you should also load the VirtualBox extension pack, located at the same URL.

- A copy of the SDN Hub All-in-One App Development Starter virtual machine image, provided on a USB device, and also available within the download section here: http://sdnhub.org/tutorials/sdn-tutorial-vm/ Of note: we are using the 64-bit version of the SDN Hub tutorial VM. If your ISP blocks peer-to-peer networking, use the direct download on the web site.

- A functioning Internet connection.

SDN Lab 1 Goals:

By completing this lab, the user will become familiar with:

1. The use of VirtualBox to launch the SDN Hub All-in-One App Development Starter virtual machine.

2. Generate a virtual network using Mininet.

3. Install the OpenDaylight controller and enable network hub functionality.

4. Validate communications between Mininet hosts.

5. Use Wireshark to verify OpenFlow communications between Mininet switches and the OpenDaylight controller.

6. Save the completed state of the virtual lab.

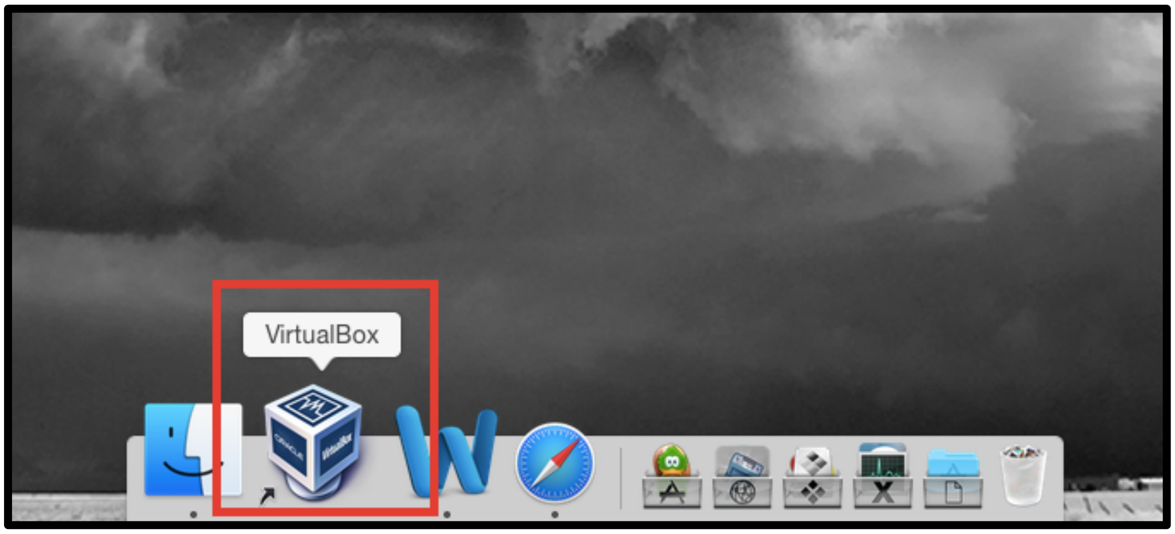
Lab Section 1.1: Installing and Launching the SDN Hub All-in-One App Development Starter virtual machine in the VirtualBox environment.

In this lab section, we will install the SDN Hub All-in-One App Development Starter virtual machine on the free Oracle VirtualBox emulator. The latest version of Oracle's Virtual Box is available via: https://www.virtualbox.org, and is supported on the majority of desktop operating systems. The SDN Hub All-in-One App Development Starter virtual machine is provided on a 16GB USB drive. Alternatively, the latest version can be downloaded here: http://sdnhub.org/tutorials/sdn-tutorial-vm/

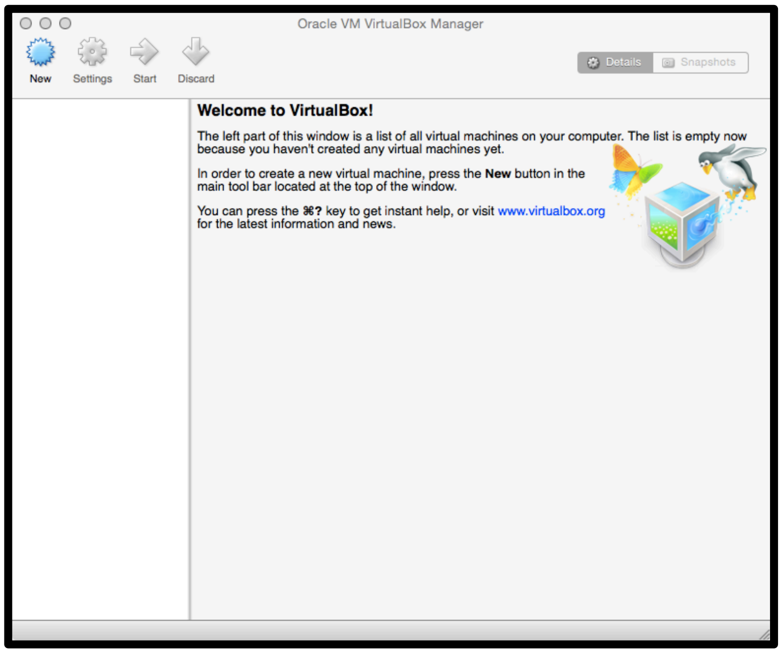
This lab section assumes that VirtualBox is already installed on your desktop environment. The VirtualBox manual, which contains installation and operation instructions, is located here: https://www.virtualbox.org/manual/UserManual.html

Step 1: Launch the VirtualBox emulator.

a. Click the VirtualBox icon:



b. You should now see the VirtualBox application window:



Welcome to VirtualBox!

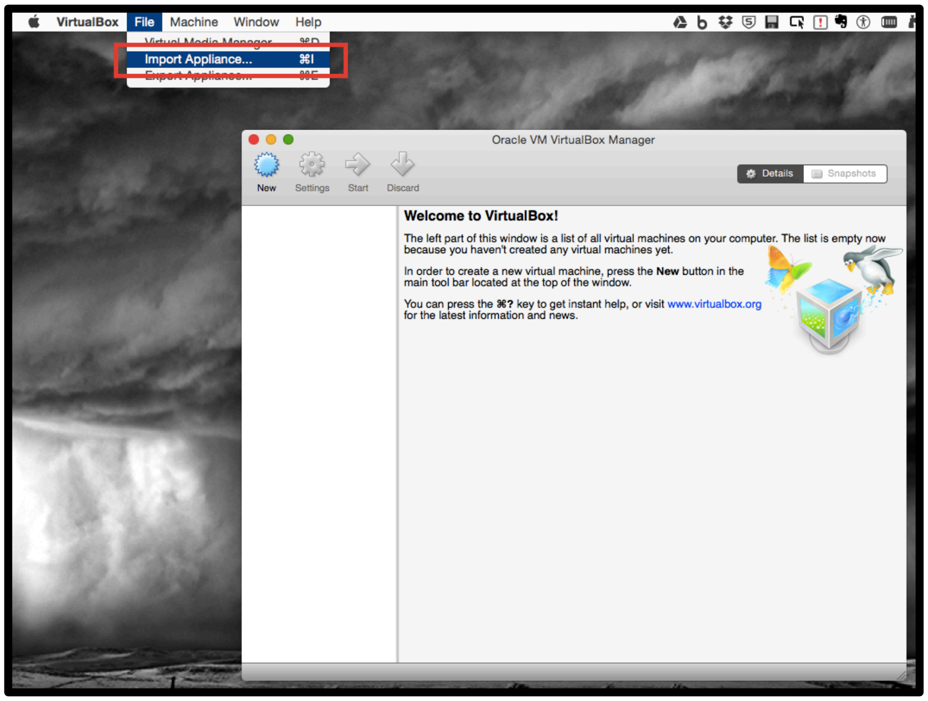
The left part of this window is a list of all virtual machines on your computer. The list is empty now because you haven't created any virtual machines yet.

In order to create a new virtual machine, press the **New** button in the main tool bar located at the top of the window.

You can press the ⌘? key to get instant help, or visit www.virtualbox.org for the latest information and news.

Step 2: Load the SDN Hub virtual Machine into the VirtualBox emulator.

a. From the VirtualBox menu, select "File > Import Appliance"



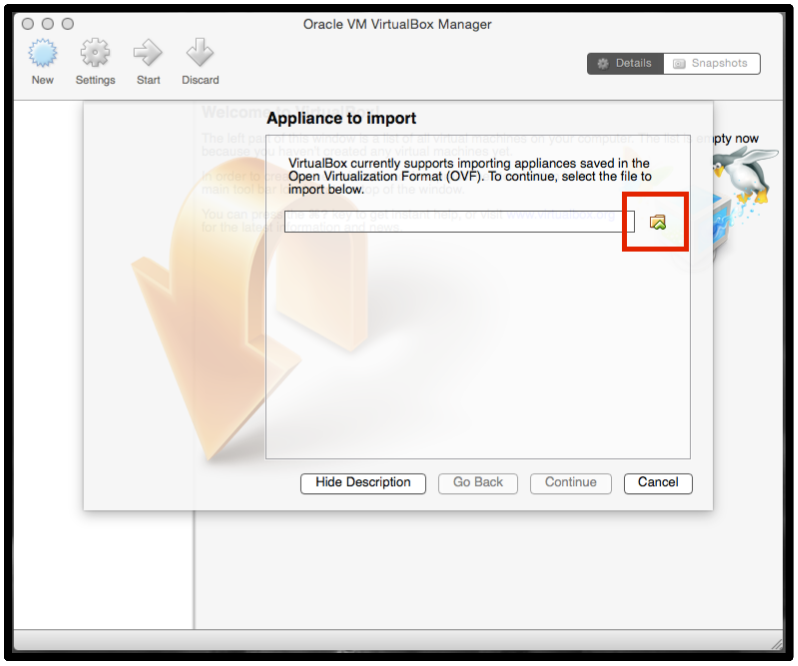
Welcome to VirtualBox!

The left part of this window is a list of all virtual machines on your computer. The list is empty now because you haven't created any virtual machines yet.

In order to create a new virtual machine, press the **New** button in the main tool bar located at the top of the window.

You can press the ⌘? key to get instant help, or visit www.virtualbox.org for the latest information and news.

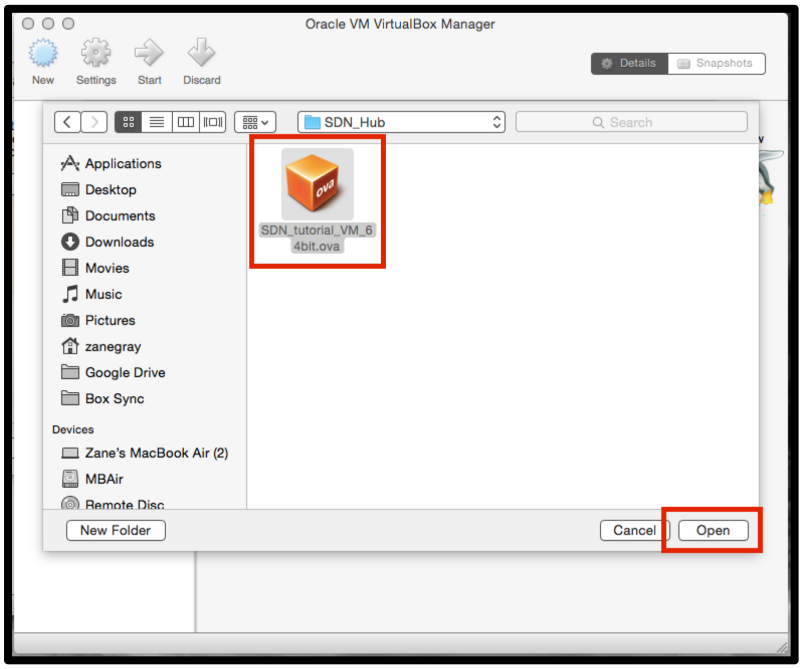
b. In the resulting window, select the folder icon to navigate to the location of your SDN Hub All-in-One App Development Starter virtual machine file.



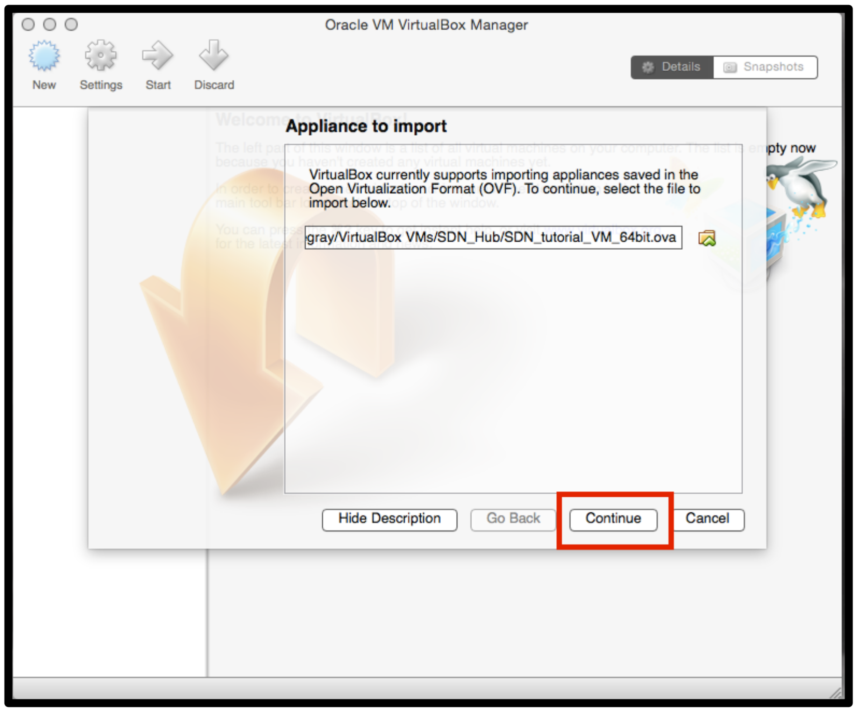
Appliance to import

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

c. From the resulting window, navigate to the SDN Hub virtual machine file. Highlight the file, and click the "Open" button.



d. Verify that the file path is correct, and click "Continue".

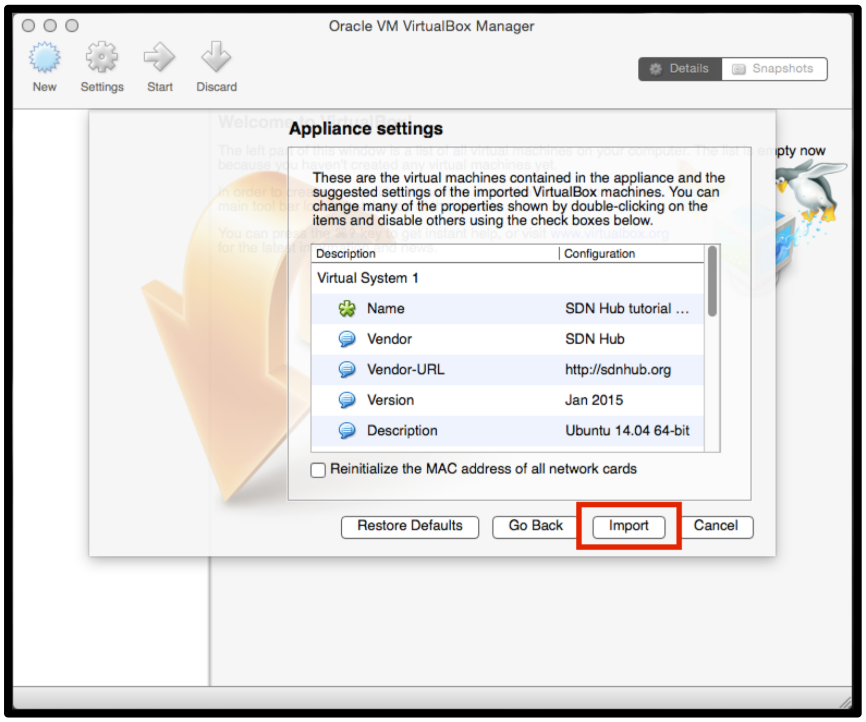


Appliance to import

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

gray/VirtualBox VMs/SDN_Hub/SDN_tutorial_VM_64bit.ova

e. From the resulting window, click "Import". For the purpose of this lab exercise, we will accept the default virtual machine settings.



Appliance settings

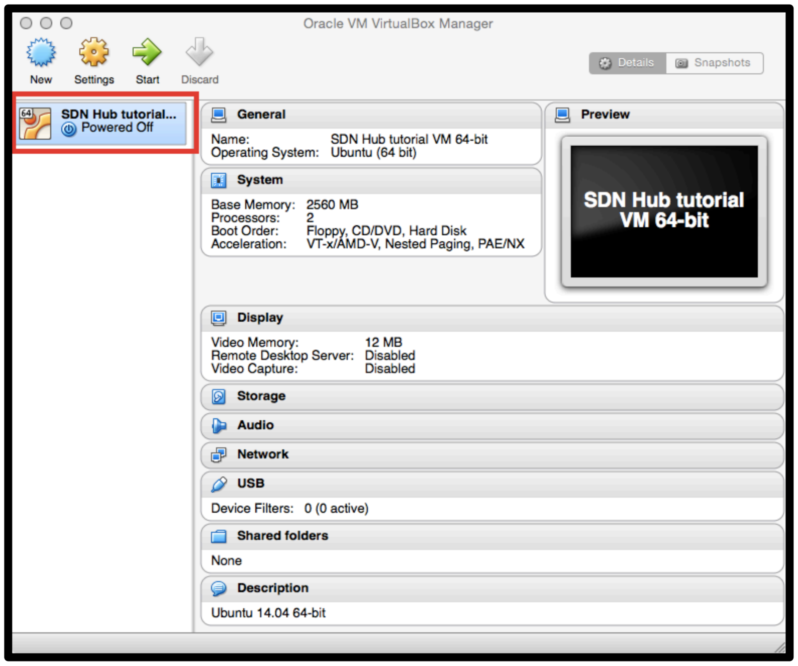
These are the virtual machines contained in the appliance and the suggested settings of the imported VirtualBox machines. You can change many of the properties shown by double-clicking on the items and disable others using the check boxes below.

Description	Configuration
Virtual System 1	
Name	SDN Hub tutorial ...
Vendor	SDN Hub
Vendor-URL	http://sdnhub.org
Version	Jan 2015
Description	Ubuntu 14.04 64-bit

☐ Reinitialize the MAC address of all network cards

Step 3: Launch the virtual machine.

a. Verify that the SDN Hub Tutorial virtual machine is now available through the VirtualBox emulator.



SDN Hub tutorial...
Powered Off

General
Name: SDN Hub tutorial VM 64-bit
Operating System: Ubuntu (64 bit)

System
Base Memory: 2560 MB
Processors: 2
Boot Order: Floppy, CD/DVD, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX

Preview

Display
Video Memory: 12 MB
Remote Desktop Server: Disabled
Video Capture: Disabled

Storage

Audio

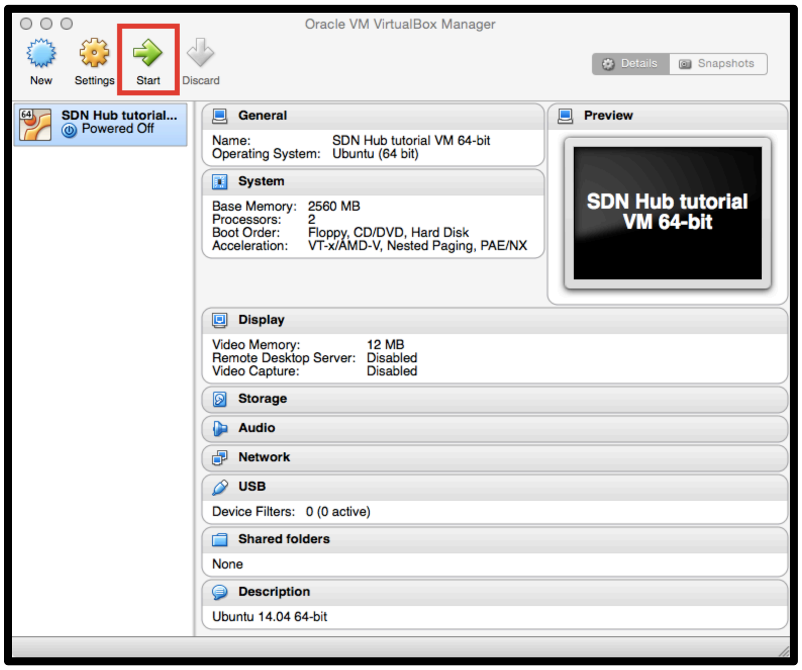
Network

USB
Device Filters: 0 (0 active)

Shared folders
None

Description
Ubuntu 14.04 64-bit

b. Press the “Start” icon to power on your virtual machine.



SDN Hub tutorial...
Powered Off

General
Name: SDN Hub tutorial VM 64-bit
Operating System: Ubuntu (64 bit)

System
Base Memory: 2560 MB
Processors: 2
Boot Order: Floppy, CD/DVD, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX

Preview

Display
Video Memory: 12 MB
Remote Desktop Server: Disabled
Video Capture: Disabled

Storage

Audio

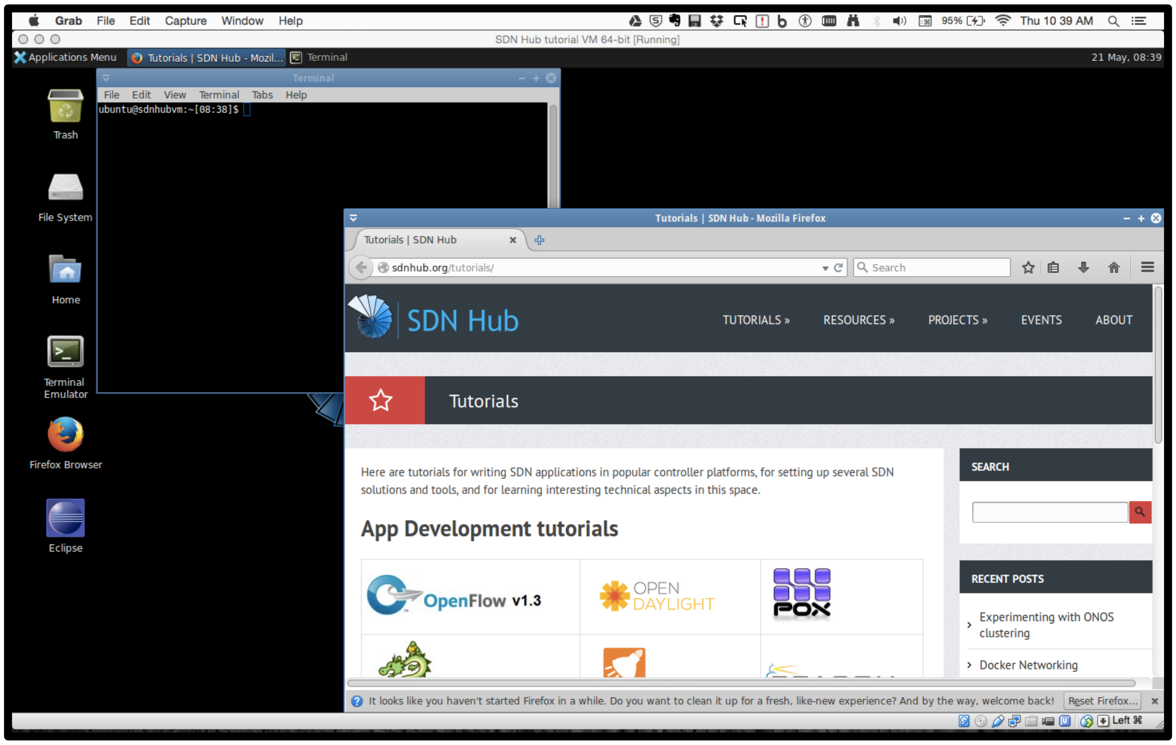
Network

USB
Device Filters: 0 (0 active)

Shared folders
None

Description
Ubuntu 14.04 64-bit

You should now see the VirtualBox BIOS screen, followed by an Ubuntu startup menu. Allow the Ubuntu installation to fully boot (should take ~30 seconds).



```
ubuntu@sdnhubvm:~[08:38]$
```

Here are tutorials for writing SDN applications in popular controller platforms, for setting up several SDN solutions and tools, and for learning interesting technical aspects in this space.

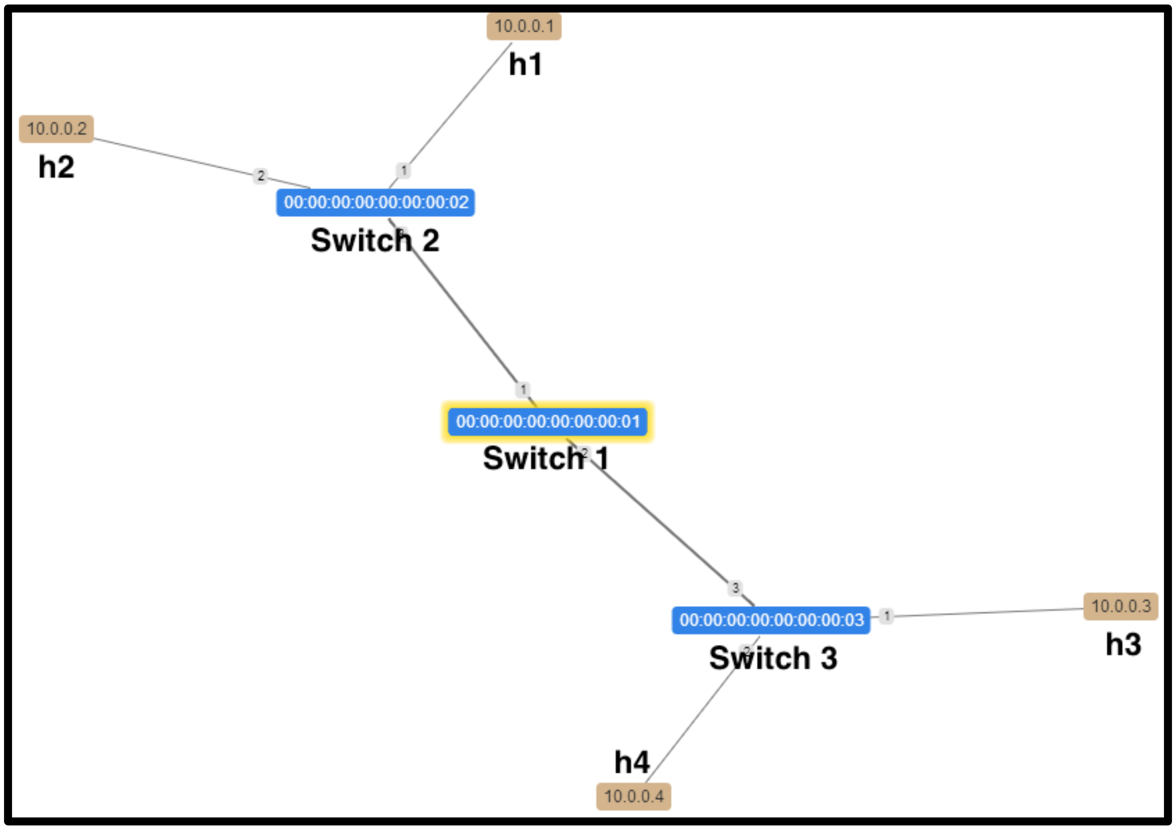
App Development tutorials

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

Congratulations! Your virtual machine is now ready to use.

Lab Section 1.2: Generate a virtual network using Mininet.

In this lab section, we will leverage a network virtualization tool called Mininet to create a small network consisting of three network switches and four network hosts. The topology will be a single core switch, and two edge switches, each containing two hosts.

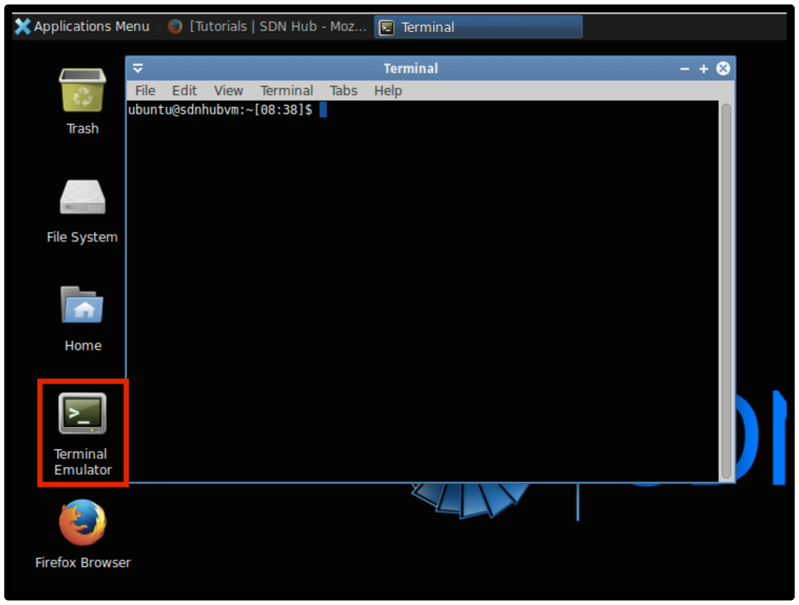


Mininet is a lightweight virtualization engine capabile of generating small networks of routers, switches, hosts, and virtual links between the devices within a single Linux environment. As such, Mininet has no graphical user interface. More detailed information on Mininet can be found here: http://mininet.org

NOTE*: *Mininet has no graphical user interface. This lab section will rely heavily on the command line interface.

Step 1: Generate the Mininet topology.

a. Open a terminal session within the SDN Hub virtual machine. The Terminal Emulator application icon can be found on the virtual machine desktop.

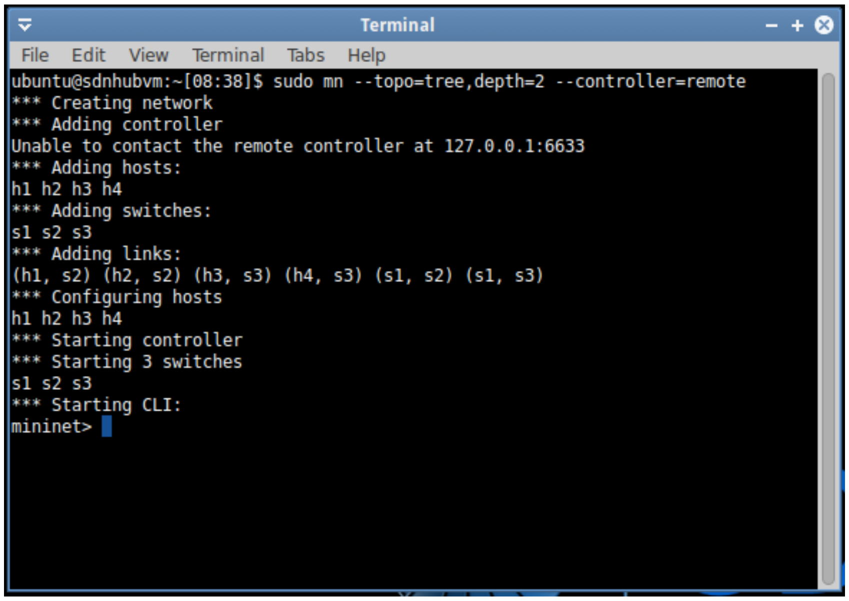


```
ubuntu@sdnhubvm:~[08:38]$
```

b. Within the terminal window, type the following command:

```
sudo mn --topo=tree,depth=2 --controller=remote
```

and press "***Enter***". Your terminal should look like this:

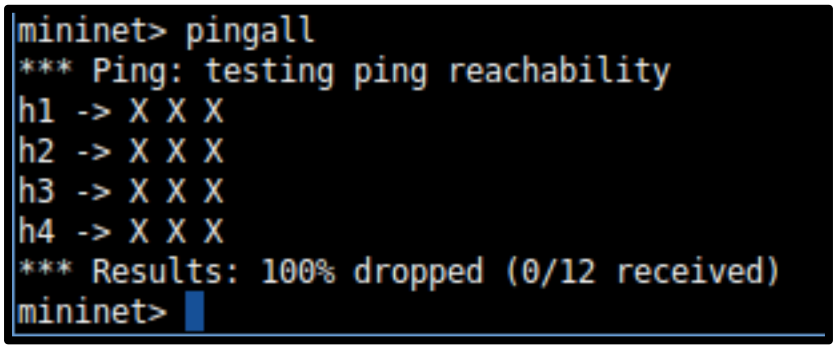


```
ubuntu@sdnhubvm:~[08:38]$ sudo mn --topo=tree,depth=2 --controller=remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3
*** Adding links:
(h1, s2) (h2, s2) (h3, s3) (h4, s3) (s1, s2) (s1, s3)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
*** Starting 3 switches
s1 s2 s3
*** Starting CLI:
mininet>
```

The command instructed Mininet to generate a "Tree" topology ("`--topo=tree`") that is two switches deep ("`depth=2`"). This provided three switches logically connected in a hub-and-spoke manner. Unless otherwise specified, Mininet attaches two hosts to each "spoke" switch in the "Tree" topology – hence our four virtual hosts.

Step 2: Test communications within the mininet.

Next, we will attempt to verify communications between our network hosts. Within the terminal window, issue the "`pingall`" command.



```
mininet> pingall
*** Ping: testing ping reachability
h1 -> X X X
h2 -> X X X
h3 -> X X X
h4 -> X X X
*** Results: 100% dropped (0/12 received)
mininet> 
```

The "`pingall`" command instructs each virtual host to send an ICMP packet request to all other hosts within the network, and record the success or failure of an ICMP packet response. In our case all of our are not receiving successful ICMP packet responses, indicated by an "X", from the other hosts within the Mininet topology.

Referring back to **Step 1**; we instructed Mininet to generate a topology of switches that require orchestration through a remote, OpenFlow controller ("`--controller=remote`"). Until we enable an OpenFlow controller, our Mininet switches will not know how to forward traffic between the hosts.

Lab Section 1.3: Installing the OpenDaylight controller.

In this lab section, we will install the OpenDaylight controller and enable network hub functionality. The overall goal is to allow host-to-host communication between our Mininet hosts within our virtualized network topology.

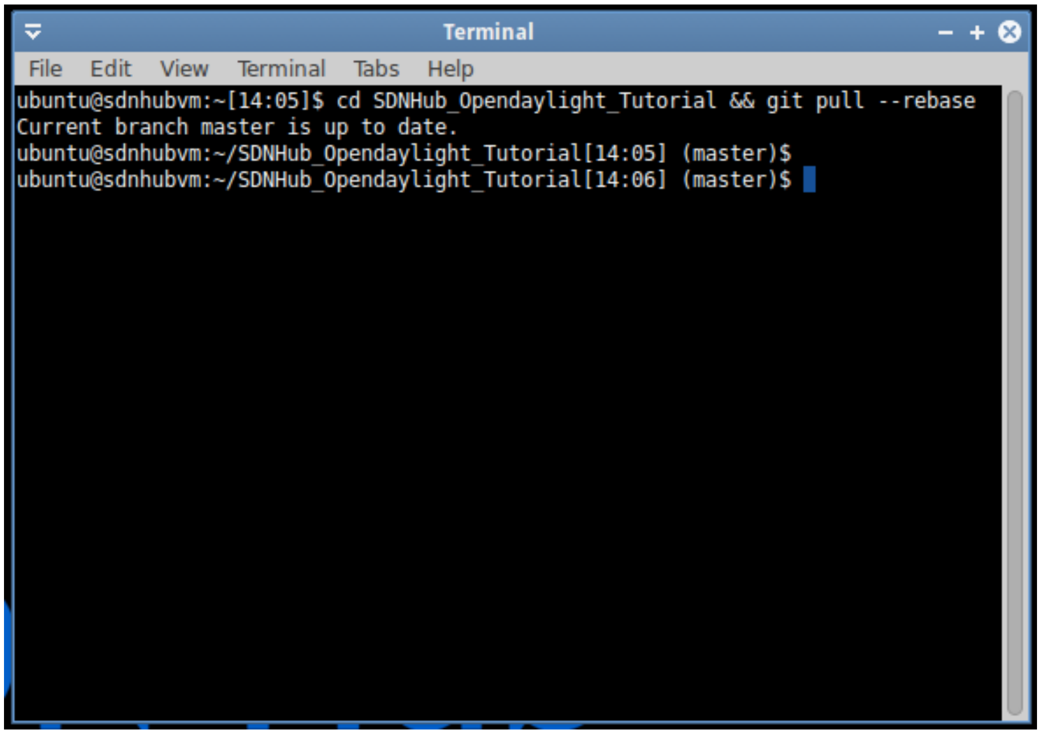
OpenDaylight is a modular, software defined networking controller. It allow for network programmability and orchestration. And, more importantly, it has a graphical user interface that we can leverage to gain insight into our virtualized network topology.

More information regarding OpenDaylight can be found here:
http://www.opendaylight.org

Step 0 [Optional – Depending on your Internet connectivity]: Update the OpenDaylight software that is included with the SDN Hub Tutorial virtual machine.

Open a new Terminal Emulator window (your Mininet is running in the existing window!), and type the following:

```
cd SDNHub_Opendaylight_Tutorial && git pull --rebase
```

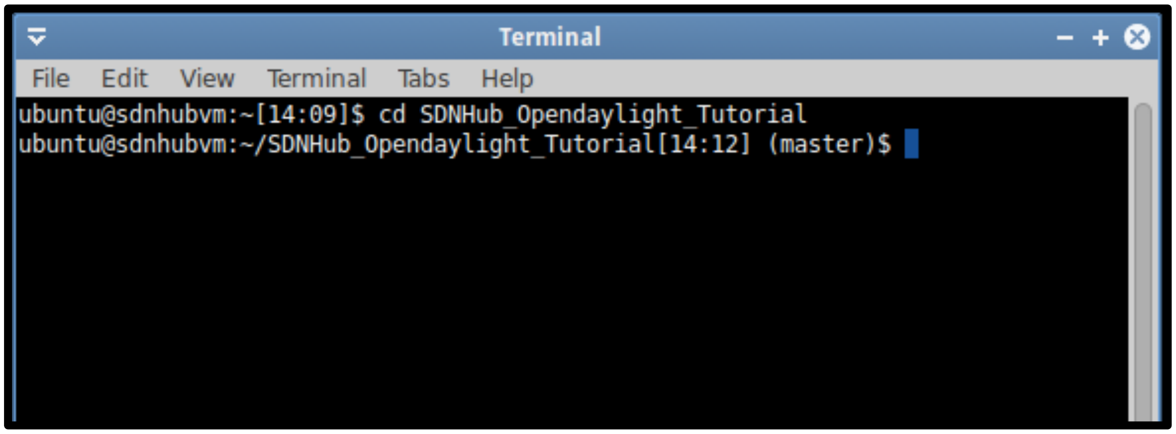


```
ubuntu@sdnhubvm:~[14:05]$ cd SDNHub_Opendaylight_Tutorial && git pull --rebase
Current branch master is up to date.
ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial[14:05] (master)$
ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial[14:06] (master)$
```

Step 2: Start OpenDaylight with basic Hub functionality.

a. Open a new Terminal Emulator window (your Mininet is running in the existing window!). Within the new window, change to the OpenDaylight directory by typing the following:

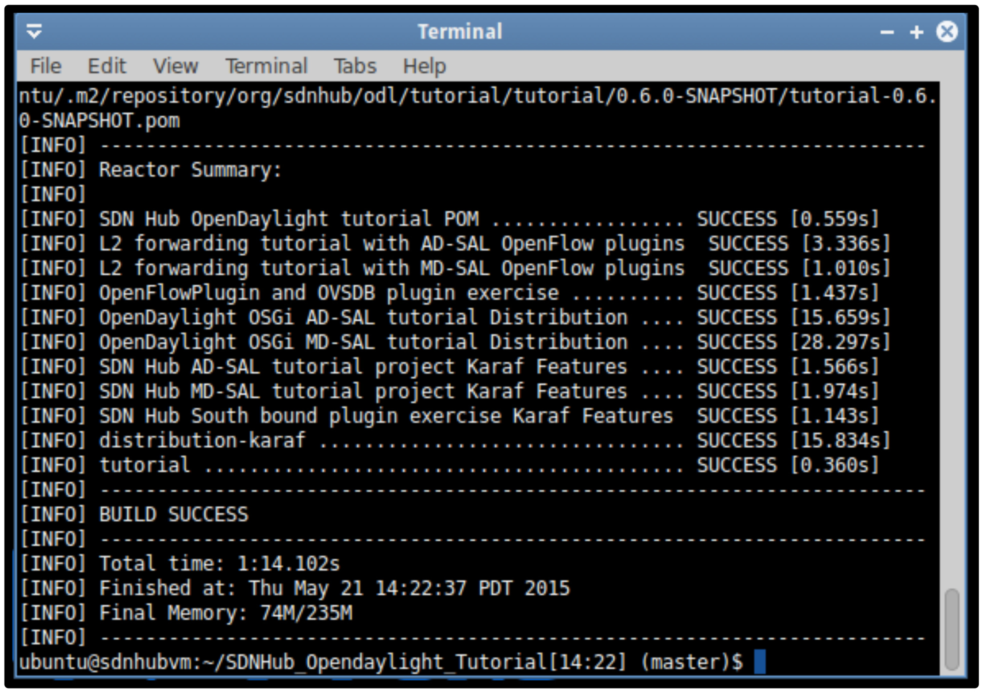
```
cd SDNHub_Opendaylight_Tutorial
```



```
ubuntu@sdnhubvm:~[14:09]$ cd SDNHub_Opendaylight_Tutorial
ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial[14:12] (master)$
```

b. Make and install the OpenDaylight software using the following command (NOTE: This step requires a working Internet connection):

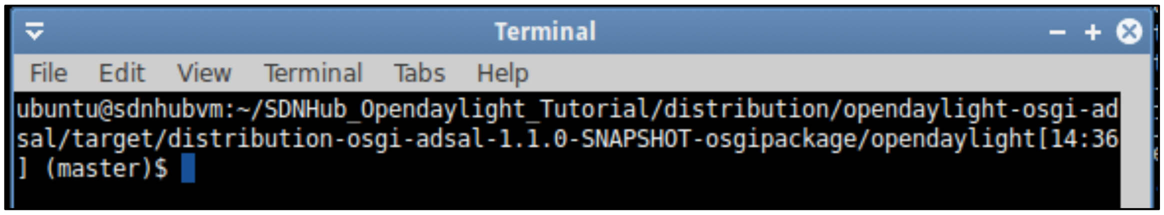
```
mvn install -nsu
```



```
ntu/.m2/repository/org/sdnhub/odl/tutorial/tutorial/0.6.0-SNAPSHOT/tutorial-0.6.
0-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] SDN Hub OpenDaylight tutorial POM ................. SUCCESS [0.559s]
[INFO] L2 forwarding tutorial with AD-SAL OpenFlow plugins  SUCCESS [3.336s]
[INFO] L2 forwarding tutorial with MD-SAL OpenFlow plugins  SUCCESS [1.010s]
[INFO] OpenFlowPlugin and OVSDB plugin exercise .......... SUCCESS [1.437s]
[INFO] OpenDaylight OSGi AD-SAL tutorial Distribution .... SUCCESS [15.659s]
[INFO] OpenDaylight OSGi MD-SAL tutorial Distribution .... SUCCESS [28.297s]
[INFO] SDN Hub AD-SAL tutorial project Karaf Features .... SUCCESS [1.566s]
[INFO] SDN Hub MD-SAL tutorial project Karaf Features .... SUCCESS [1.974s]
[INFO] SDN Hub South bound plugin exercise Karaf Features  SUCCESS [1.143s]
[INFO] distribution-karaf ................................ SUCCESS [15.834s]
[INFO] tutorial .......................................... SUCCESS [0.360s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 1:14.102s
[INFO] Finished at: Thu May 21 14:22:37 PDT 2015
[INFO] Final Memory: 74M/235M
[INFO] ------------------------------------------------------------------------
ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial[14:22] (master)$
```

c. Open another Terminal Emulator window (this should be the third one on your virtual desktop) and change to the OpenDaylight directory using the following command (NOTE: Type this as a single line of text, and there is a space after "cd"):

```
cd
~/SDNHub_Opendaylight_Tutorial/distribution/opendaylight-osgi-adsal/target/distribution-osgi-adsal-1.1.0-SNAPSHOT-osgipackage/opendaylight
```

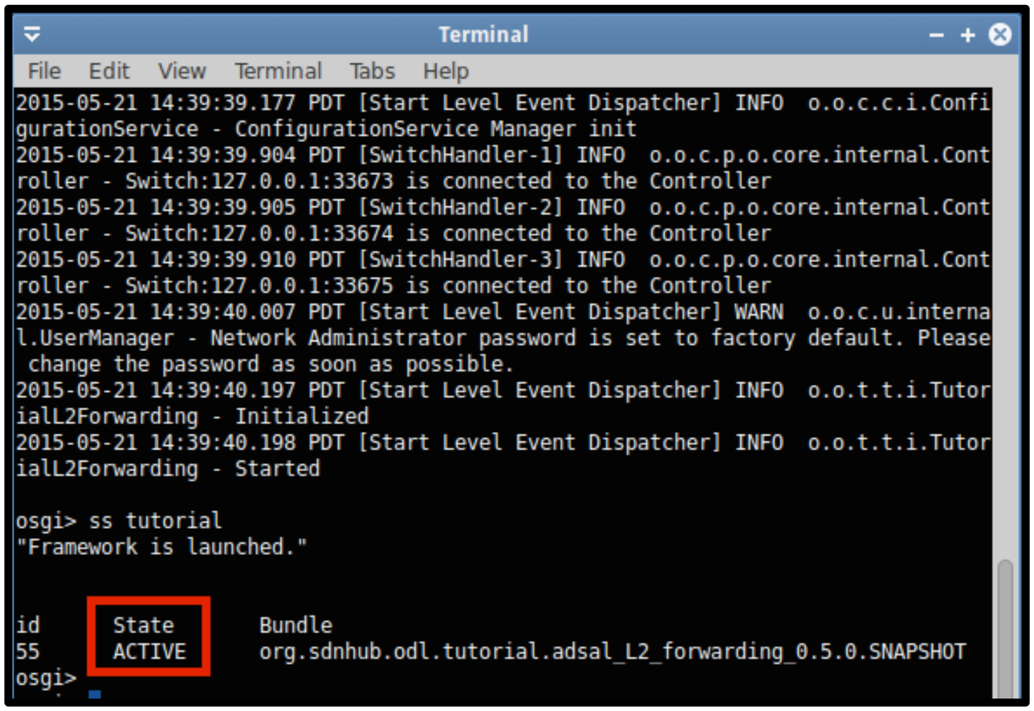


```
ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial/distribution/opendaylight-osgi-ad
sal/target/distribution-osgi-adsal-1.1.0-SNAPSHOT-osgipackage/opendaylight[14:36
] (master)$
```

d. Now, start the OpenDaylight controller tutorial by running the following command within the same Terminal Emulator window:

```
./run.sh
```

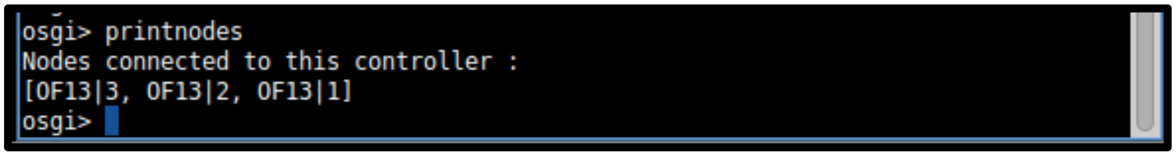
Verify that the controller process is running by typing "***ss tutorial***" in the terminal window, and look for an "Active" process status.



```
2015-05-21 14:39:39.177 PDT [Start Level Event Dispatcher] INFO  o.o.c.c.i.Confi
gurationService - ConfigurationService Manager init
2015-05-21 14:39:39.904 PDT [SwitchHandler-1] INFO  o.o.c.p.o.core.internal.Cont
roller - Switch:127.0.0.1:33673 is connected to the Controller
2015-05-21 14:39:39.905 PDT [SwitchHandler-2] INFO  o.o.c.p.o.core.internal.Cont
roller - Switch:127.0.0.1:33674 is connected to the Controller
2015-05-21 14:39:39.910 PDT [SwitchHandler-3] INFO  o.o.c.p.o.core.internal.Cont
roller - Switch:127.0.0.1:33675 is connected to the Controller
2015-05-21 14:39:40.007 PDT [Start Level Event Dispatcher] WARN  o.o.c.u.interna
l.UserManager - Network Administrator password is set to factory default. Please
 change the password as soon as possible.
2015-05-21 14:39:40.197 PDT [Start Level Event Dispatcher] INFO  o.o.t.t.i.Tutor
ialL2Forwarding - Initialized
2015-05-21 14:39:40.198 PDT [Start Level Event Dispatcher] INFO  o.o.t.t.i.Tutor
ialL2Forwarding - Started

osgi> ss tutorial
"Framework is launched."


id      State       Bundle
55      ACTIVE      org.sdnhub.odl.tutorial.adsal_L2_forwarding_0.5.0.SNAPSHOT
osgi>
```

Step 2: Verify that the Mininet switches are communicating with the controller.

Issue the "***printnodes***" command at the OSGI terminal prompt.

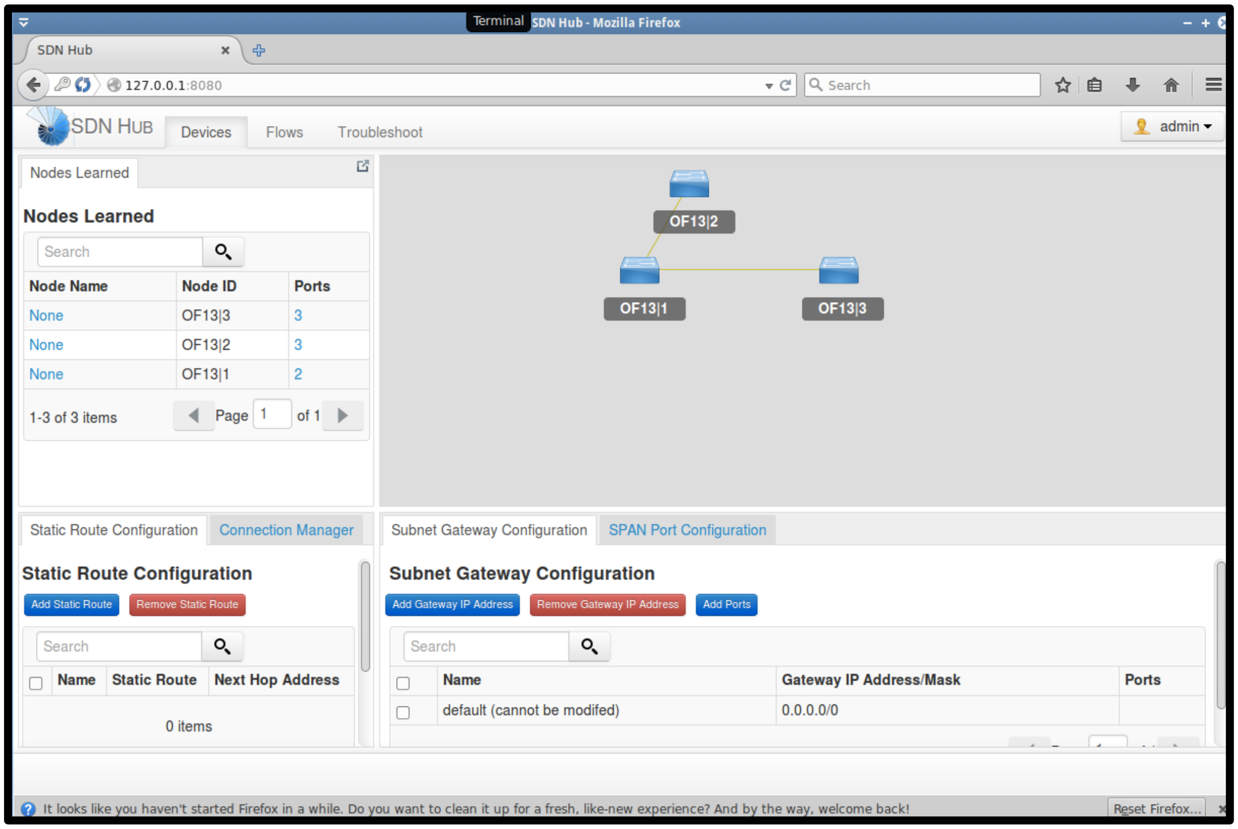


```
osgi> printnodes
Nodes connected to this controller :
[OF13|3, OF13|2, OF13|1]
osgi>
```

Reviewing the output, we can see three switches connected. Of note: "OF13" implies that the switches are connected using OpenFlow version 1.3 versus the legacy 1.0 version.

Step 3: View the logical topology.

Open a Firefox window withing the SDN Hub Tutorial virtual machine, and browse to http://127.0.0.1:8080. (NOTE: The username and password are both '**admin**').



Nodes Learned

Node Name	Node ID	Ports
None	OF13|3	3
None	OF13|2	3
None	OF13|1	2

1-3 of 3 items Page 1 of 1

Static Route Configuration

Name	Static Route	Next Hop Address

0 items

Subnet Gateway Configuration

Name	Gateway IP Address/Mask	Ports
default (cannot be modifed)	0.0.0.0/0	

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

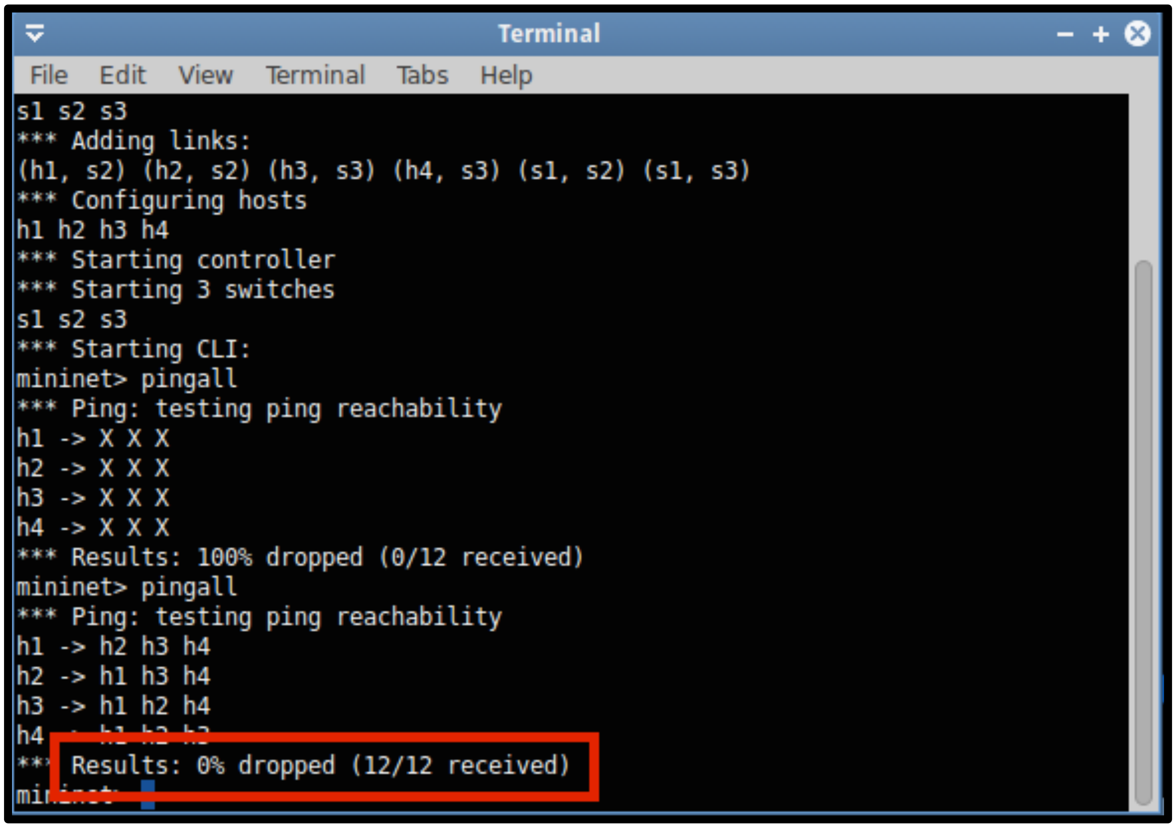
Congratulations! You now have an OpenDaylight controller attached to three network switches.

Lab Section 1.4: Validating communications between hosts.

In this lab section, we will verify the "hub" functionality that connecting our OpenDaylight controller is providing for our Mininet switches. We will use the ping command to verify communications between our Mininet hosts within our virtualized network topology.

This lab section will refer back to Lab **Section 1.2: Step 2**, in which we used the Mininet "pingall" command.

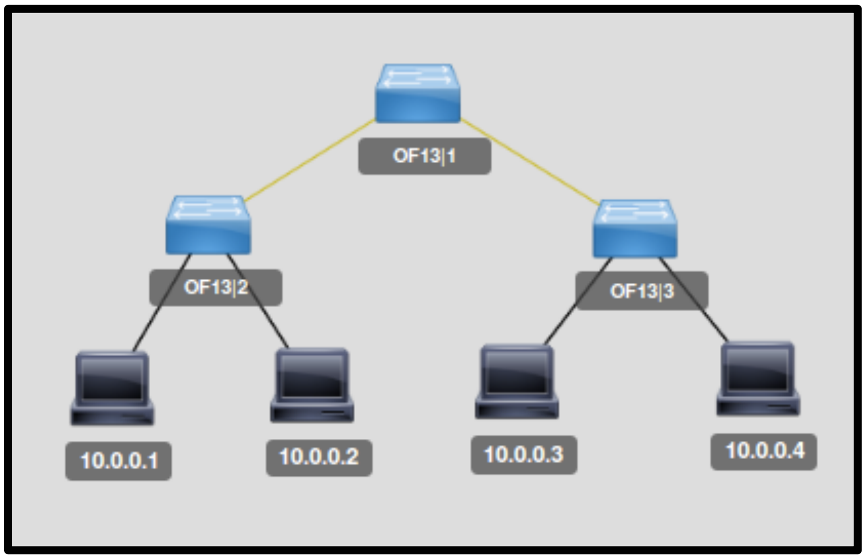
Step 1: Select the Terminal Emulator window with the "***mininet>***" prompt. Reissue the "`pingall`" command:



```
s1 s2 s3
*** Adding links:
(h1, s2) (h2, s2) (h3, s3) (h4, s3) (s1, s2) (s1, s3)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
*** Starting 3 switches
s1 s2 s3
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1 -> X X X
h2 -> X X X
h3 -> X X X
h4 -> X X X
*** Results: 100% dropped (0/12 received)
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet>
```

We now have full communications between our Mininet hosts.

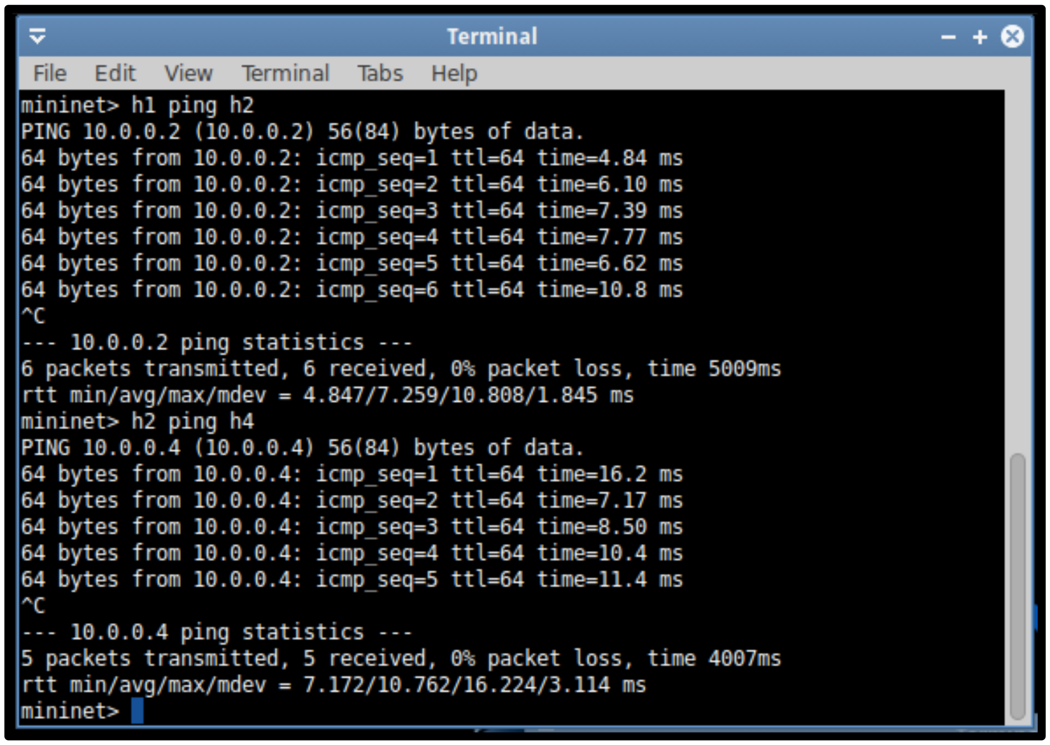
Step 2: Return to your Firefox browser window. Reload the controller web page. Note that the network topology map has now changed to include the hosts:



Step 3 [BONUS SECTION]: Optimize communications by enabling network switching functionality.

Before proceeding with this section, you may want to visit and complete Lab Section 1.6 of this manual. The following section makes changes to Java files that are not trivial to reverse. In the event that this section is unsuccessful, you may need to "roll back" your virtual machine to a functional snapshot.

a. Use the ping command within your Mininet terminal window to verify communications between individual hosts. For example: "`h1 ping h3`" will request virtual host 1 to send ICMP packets to virtual host 3. Issue multiple commands to determine response between the various hosts. Use the "***CTRL + C***" key sequence to stop the ping traffic after four or five successive ICMP replies:



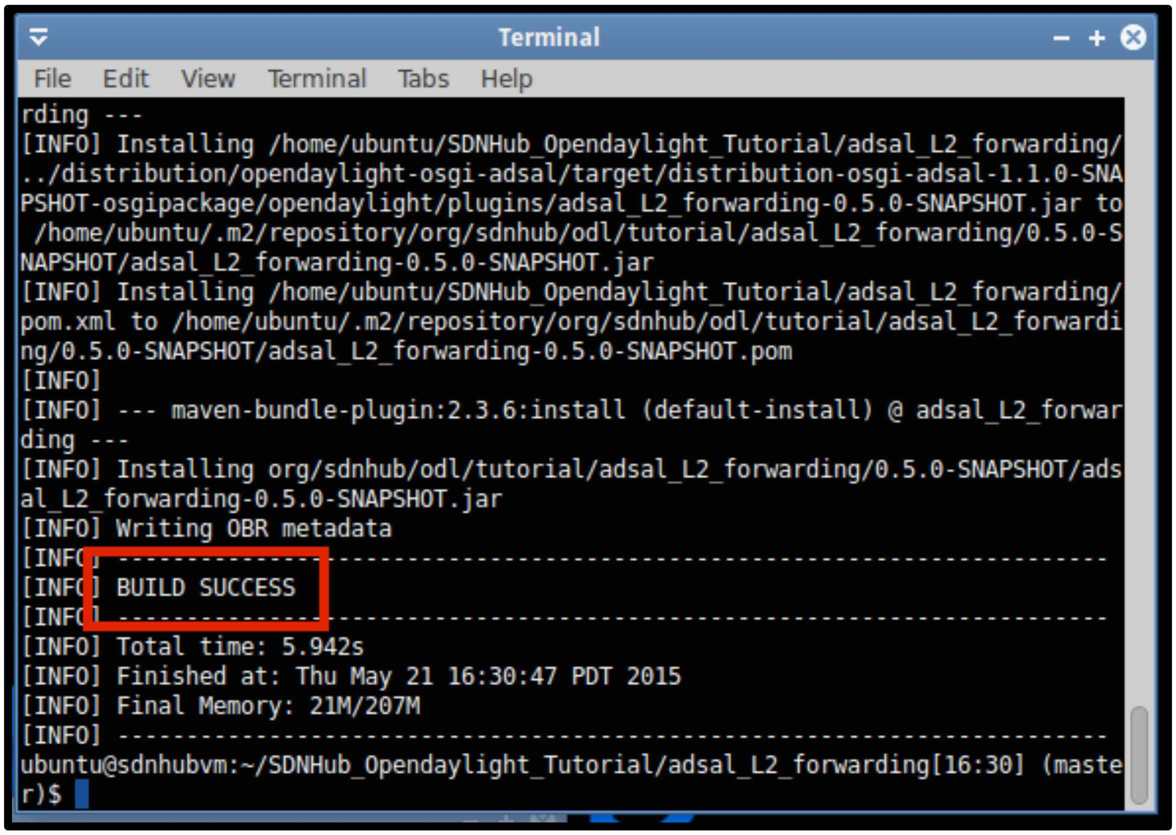
```
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=4.84 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=6.10 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=7.39 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=7.77 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=6.62 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=10.8 ms
^C
--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 4.847/7.259/10.808/1.845 ms
mininet> h2 ping h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=16.2 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=7.17 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=8.50 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=10.4 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=11.4 ms
^C
--- 10.0.0.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 7.172/10.762/16.224/3.114 ms
mininet>
```

Compared to the "***pingall***" command, issuing the ping command between individual hosts will measure the latency across our network paths.

b. Open a new terminal window, and issue the following commands. NOTE: Each section of commands represents a single line with no line breaks.

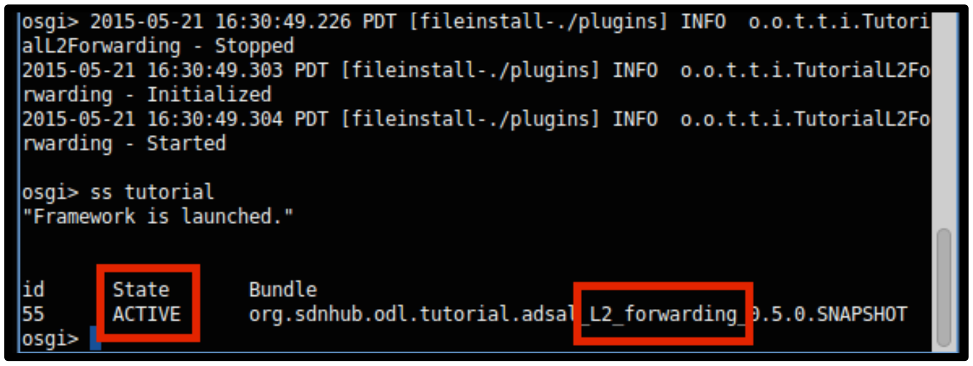
```
cd SDNHub_Opendaylight_Tutorial

sed -I -e "s/function = \"hub\"/function =
\"switch\"/g" `find . -name TutorialL2Forwarding.java`

cd adsal_L2_forwarding

mvn install -DskipTests -DskipIT -nsu
```

c. Verify that your terminal window displayed a successful build:



```
rding ---
[INFO] Installing /home/ubuntu/SDNHub_Opendaylight_Tutorial/adsal_L2_forwarding/
../distribution/opendaylight-osgi-adsal/target/distribution-osgi-adsal-1.1.0-SNA
PSHOT-osgipackage/opendaylight/plugins/adsal_L2_forwarding-0.5.0-SNAPSHOT.jar to
 /home/ubuntu/.m2/repository/org/sdnhub/odl/tutorial/adsal_L2_forwarding/0.5.0-S
NAPSHOT/adsal_L2_forwarding-0.5.0-SNAPSHOT.jar
[INFO] Installing /home/ubuntu/SDNHub_Opendaylight_Tutorial/adsal_L2_forwarding/
pom.xml to /home/ubuntu/.m2/repository/org/sdnhub/odl/tutorial/adsal_L2_forwardi
ng/0.5.0-SNAPSHOT/adsal_L2_forwarding-0.5.0-SNAPSHOT.pom
[INFO]
[INFO] --- maven-bundle-plugin:2.3.6:install (default-install) @ adsal_L2_forwar
ding ---
[INFO] Installing org/sdnhub/odl/tutorial/adsal_L2_forwarding/0.5.0-SNAPSHOT/ads
al_L2_forwarding-0.5.0-SNAPSHOT.jar
[INFO] Writing OBR metadata
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 5.942s
[INFO] Finished at: Thu May 21 16:30:47 PDT 2015
[INFO] Final Memory: 21M/207M
[INFO] ------------------------------------------------------------------------
ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial/adsal_L2_forwarding[16:30] (maste
r)$
```

d. Within your OSGI terminal window, (the one with the "***osgi>***" prompt) use the "***ss tutorial***" command to verify that layer-2 forwarding (network switching) is now active:



```
osgi> 2015-05-21 16:30:49.226 PDT [fileinstall-./plugins] INFO  o.o.t.t.i.Tutori
alL2Forwarding - Stopped
2015-05-21 16:30:49.303 PDT [fileinstall-./plugins] INFO  o.o.t.t.i.TutorialL2Fo
rwarding - Initialized
2015-05-21 16:30:49.304 PDT [fileinstall-./plugins] INFO  o.o.t.t.i.TutorialL2Fo
rwarding - Started

osgi> ss tutorial
"Framework is launched."


id      State       Bundle
55      ACTIVE      org.sdnhub.odl.tutorial.adsal_L2_forwarding_0.5.0.SNAPSHOT
osgi>
```

e. Within the Mininet terminal window (the one with the "***mininet>***" prompt), reissue your ping commands between hosts. Compare the results with those previously displayed when running the Mininet topology as a "hub" instead of a "switch". Your overall ping times between virtual hosts should decrease with L2 switching.

Lab Section 1.5: View OpenFlow Communications with Wireshark.

In this lab section, we will Validate OpenFlow communications between Mininet switches and the OpenDaylight controller by leveraging the WireShark packet capturing utility.

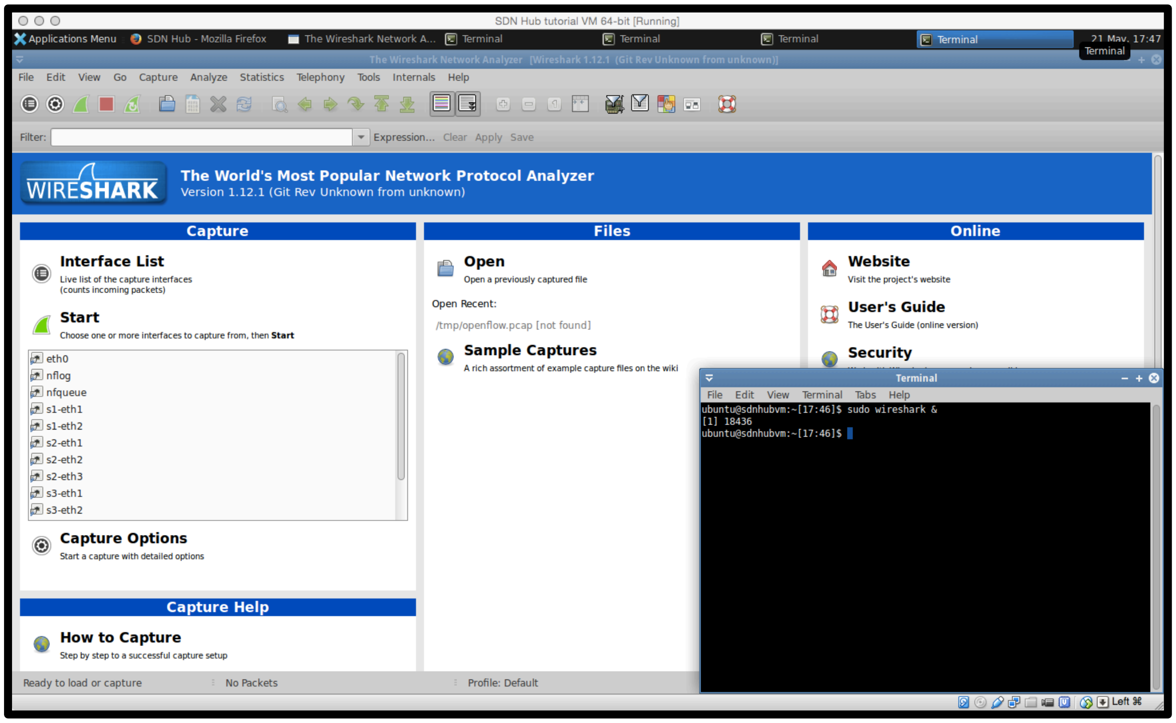
Wireshark is a packet capture utility that can be used to view network traffic, as well as decode network communications. More information on Wireshark can be found at: http://wireshark.org

Step 1: Open a third Terminal Emulator window.

Step 2: Within the new window, type the following command:

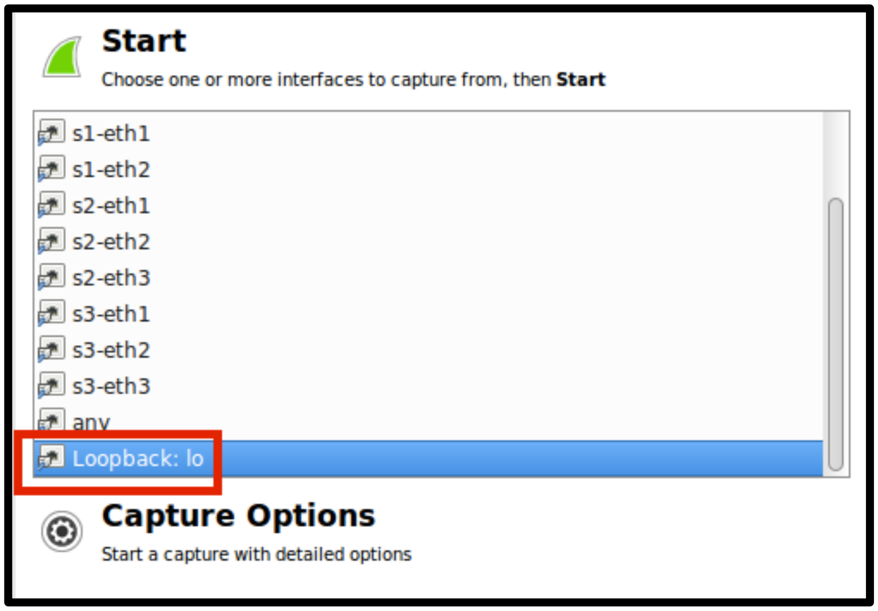
```
sudo wireshark &
```

A Wireshark application window should now open:

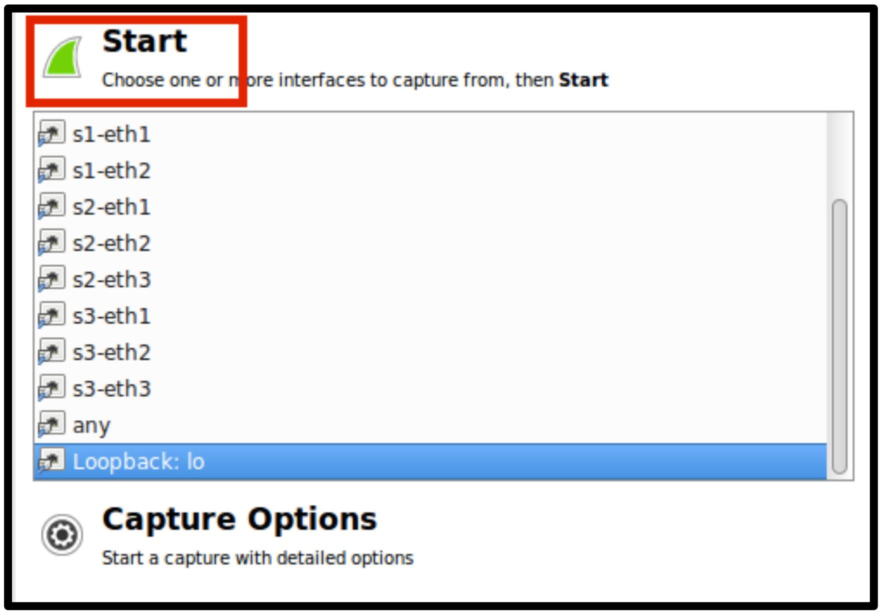

SDN Hub tutorial VM 64-bit [Running]
Applications Menu
SDN Hub - Mozilla Firefox
The Wireshark Network A...
Terminal
21 May, 17:47
The Wireshark Network Analyzer [Wireshark 1.12.1 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save
WIRESHARK
The World's Most Popular Network Protocol Analyzer
Version 1.12.1 (Git Rev Unknown from unknown)
Capture
Interface List
Live list of the capture interfaces (counts incoming packets)
Start
Choose one or more interfaces to capture from, then Start
eth0
nflog
nfqueue
s1-eth1
s1-eth2
s2-eth1
s2-eth2
s2-eth3
s3-eth1
s3-eth2
Capture Options
Start a capture with detailed options
Capture Help
How to Capture
Step by step to a successful capture setup
Files
Open
Open a previously captured file
Open Recent:
/tmp/openflow.pcap [not found]
Sample Captures
A rich assortment of example capture files on the wiki
Online
Website
Visit the project's website
User's Guide
The User's Guide (online version)
Security
Terminal
File Edit View Terminal Tabs Help
Ready to load or capture
No Packets
Profile: Default
Left ⌘

```
ubuntu@sdnhubvm:~[17:46]$ sudo wireshark &
[1] 18436
ubuntu@sdnhubvm:~[17:46]$
```

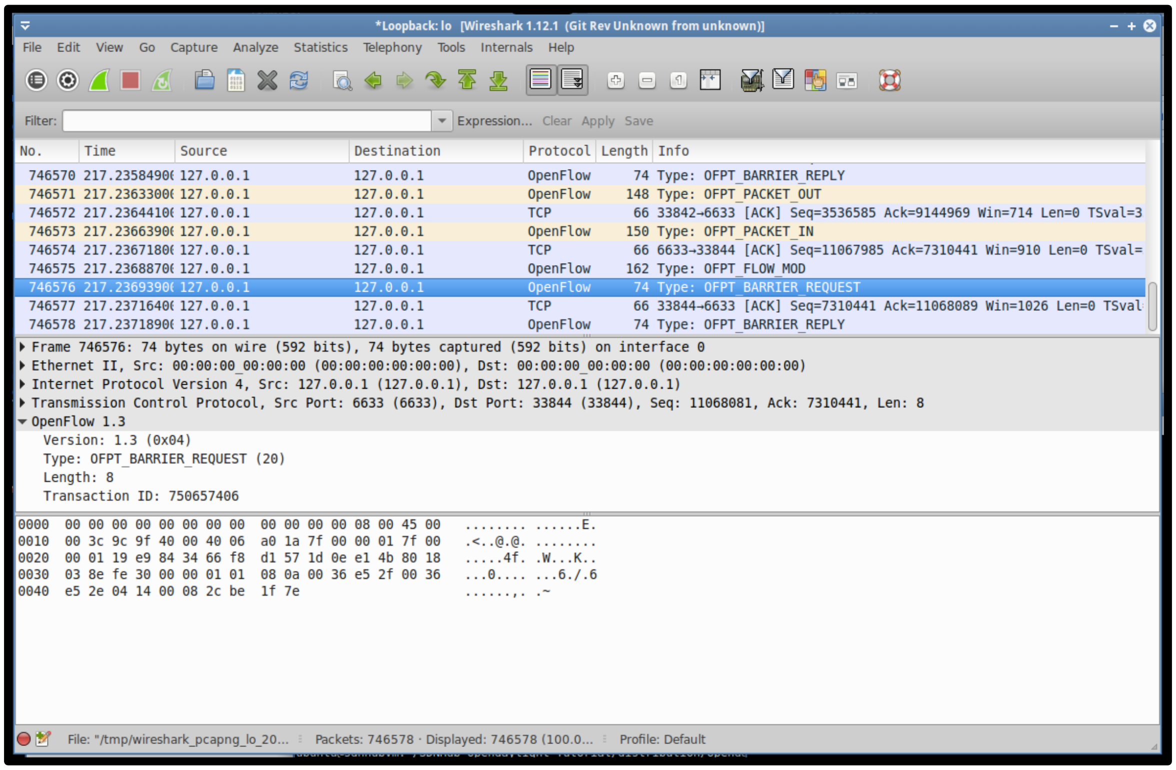
Step 3: Within the "Capture" column of the Wireshark application, select the "***Loopback: l0***" interface:


Start
Choose one or more interfaces to capture from, then **Start**
s1-eth1
s1-eth2
s2-eth1
s2-eth2
s2-eth3
s3-eth1
s3-eth2
s3-eth3
any
Loopback: lo
Capture Options
Start a capture with detailed options

Step 4: Press the "***Start***" button:


Start
Choose one or more interfaces to capture from, then **Start**
s1-eth1
s1-eth2
s2-eth1
s2-eth2
s2-eth3
s3-eth1
s3-eth2
s3-eth3
any
Loopback: lo
Capture Options
Start a capture with detailed options

You should now see a new window depicting the actual OpenFlow packets being sent to the OpenDaylight controller from the Mininet virtual network:


*Loopback: lo [Wireshark 1.12.1 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: Expression... Clear Apply Save

```
No.    Time         Source      Destination  Protocol Length Info
746570 217.2358490( 127.0.0.1   127.0.0.1    OpenFlow     74 Type: OFPT_BARRIER_REPLY
746571 217.2363300( 127.0.0.1   127.0.0.1    OpenFlow    148 Type: OFPT_PACKET_OUT
746572 217.2364410( 127.0.0.1   127.0.0.1    TCP          66 33842→6633 [ACK] Seq=3536585 Ack=9144969 Win=714 Len=0 TSval=3
746573 217.2366390( 127.0.0.1   127.0.0.1    OpenFlow    150 Type: OFPT_PACKET_IN
746574 217.2367180( 127.0.0.1   127.0.0.1    TCP          66 6633→33844 [ACK] Seq=11067985 Ack=7310441 Win=910 Len=0 TSval=
746575 217.2368870( 127.0.0.1   127.0.0.1    OpenFlow    162 Type: OFPT_FLOW_MOD
746576 217.2369390( 127.0.0.1   127.0.0.1    OpenFlow     74 Type: OFPT_BARRIER_REQUEST
746577 217.2371640( 127.0.0.1   127.0.0.1    TCP          66 33844→6633 [ACK] Seq=7310441 Ack=11068089 Win=1026 Len=0 TSval
746578 217.2371890( 127.0.0.1   127.0.0.1    OpenFlow     74 Type: OFPT_BARRIER_REPLY
```

```
▸ Frame 746576: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▸ Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▸ Transmission Control Protocol, Src Port: 6633 (6633), Dst Port: 33844 (33844), Seq: 11068081, Ack: 7310441, Len: 8
▾ OpenFlow 1.3
    Version: 1.3 (0x04)
    Type: OFPT_BARRIER_REQUEST (20)
    Length: 8
    Transaction ID: 750657406
```

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ........ ......E.
0010  00 3c 9c 9f 40 00 40 06  a0 1a 7f 00 00 01 7f 00   .<..@.@. ........
0020  00 01 19 e9 84 34 66 f8  d1 57 1d 0e e1 4b 80 18   .....4f. .W...K..
0030  03 8e fe 30 00 00 01 01  08 0a 00 36 e5 2f 00 36   ...0.... ...6./.6
0040  e5 2e 04 14 00 08 2c be  1f 7e                     ......,. .~
```

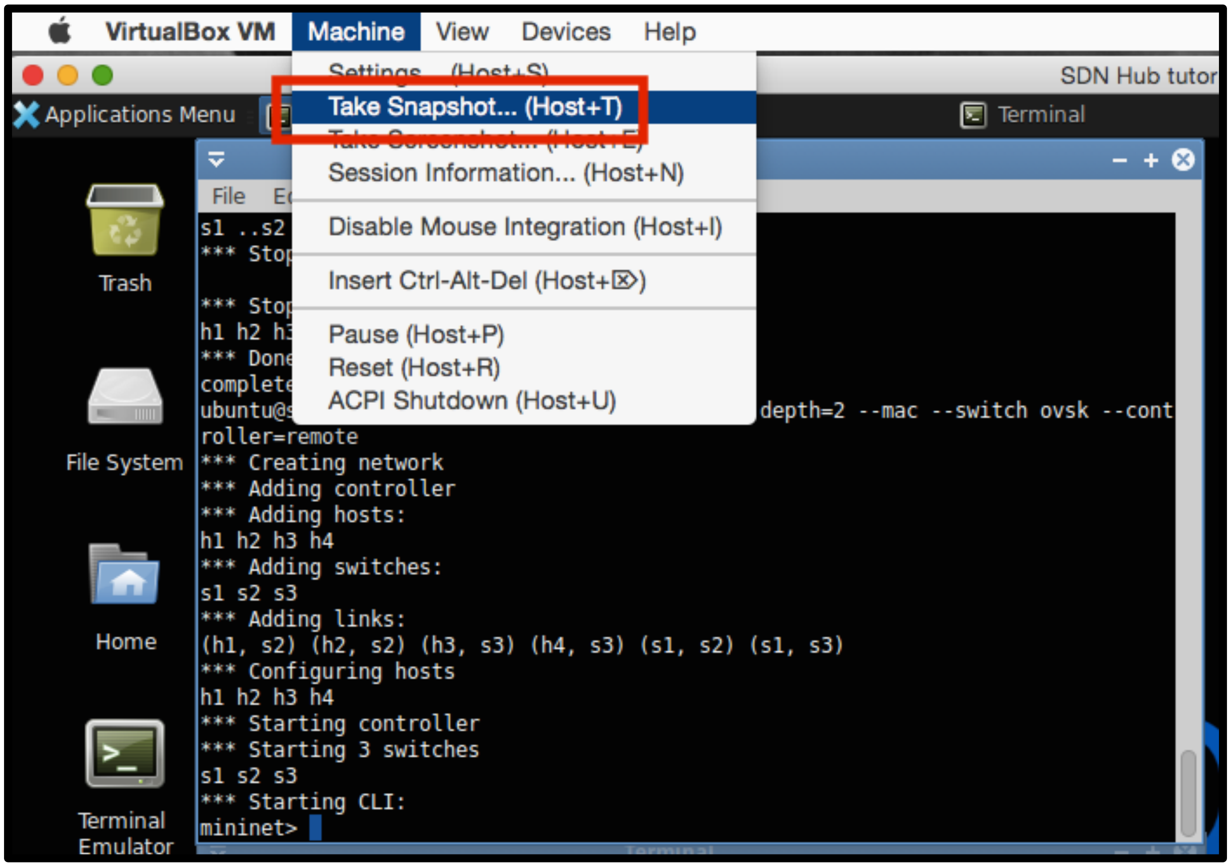
File: "/tmp/wireshark_pcapng_lo_20... Packets: 746578 · Displayed: 746578 (100.0... Profile: Default

Congratulations! You are now viewing native OpenFlow packets.

Lab Section 1.6: Saving your SDN Hub Tutorial

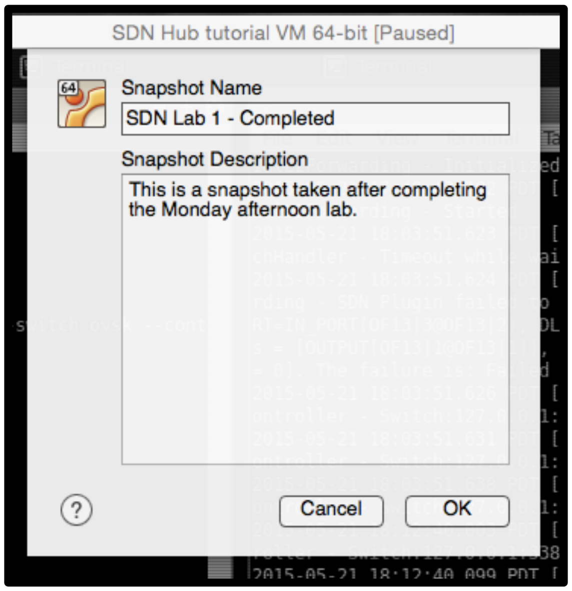
In this lab section, we will use the snapshot capabilities of VirtualBox to backup the current state of our SDN Hub Tutorial virtual machine. SDN Lab 2 will build upon this successfully completed lab, and saving the current machine state will allows us to have a "known working" lab environment.

Step 1: From the VirtualBox menu, select "***Machine > Take Snapshot…***"



```
s1 ..s2
*** Stop
*** Stop
h1 h2 h3
*** Done
complete
ubuntu@s                                   depth=2 --mac --switch ovsk --cont
roller=remote
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3
*** Adding links:
(h1, s2) (h2, s2) (h3, s3) (h4, s3) (s1, s2) (s1, s3)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
*** Starting 3 switches
s1 s2 s3
*** Starting CLI:
mininet>
```

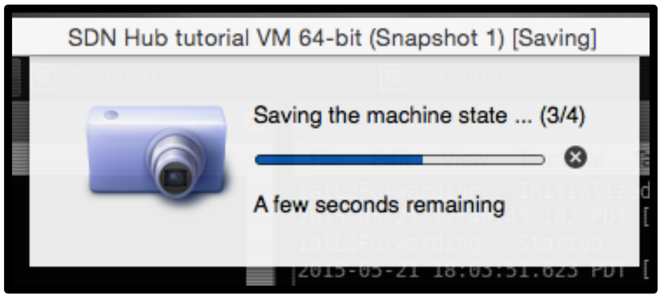
Step 2: Enter a snapshot name and description (optional) into the resulting dialog box. Note that your virtual machine is now paused:



Snapshot Name: SDN Lab 1 - Completed

Snapshot Description: This is a snapshot taken after completing the Monday afternoon lab.

Step 3: Select the "***OK***" button. You will now receive a status window that depicts the progress of the snapshot:



Saving the machine state ... (3/4)

A few seconds remaining

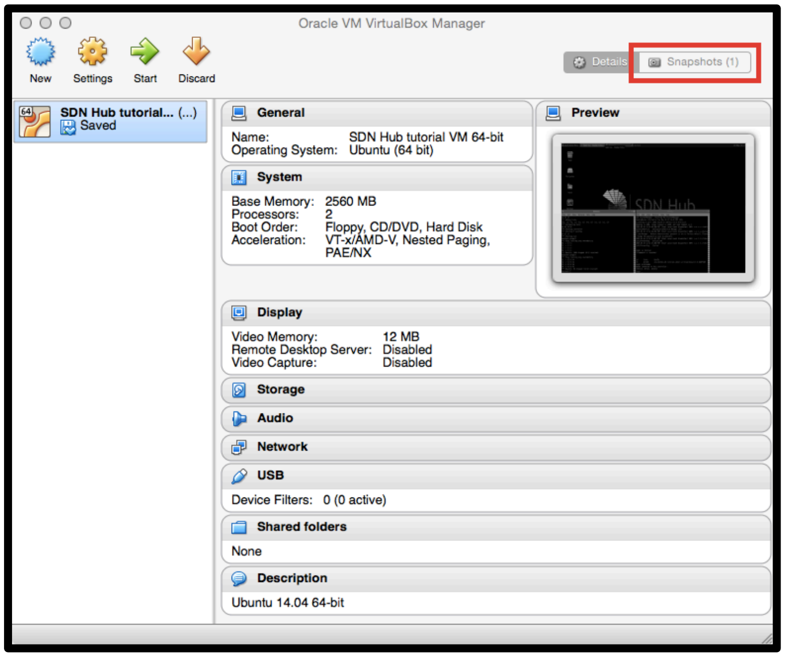
Once the snapshot process completes, your virtual machine will resume operations.

Congratulations! You have backed up your virtual machine environment, and are prepared for the second part of the SDN Lab.

Restoring from a snapshot:

In the event that you need to recover your snapshot copy and restore your Virtual Machine to a specific state, you can do so through the following steps:

Step 1: Within your VirtualBox application window, highlight your SDN Tutorial VM, and click on the "*Snapshots*" button:



General
Name: SDN Hub tutorial VM 64-bit
Operating System: Ubuntu (64 bit)

System
Base Memory: 2560 MB
Processors: 2
Boot Order: Floppy, CD/DVD, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX

Display
Video Memory: 12 MB
Remote Desktop Server: Disabled
Video Capture: Disabled

Storage

Audio

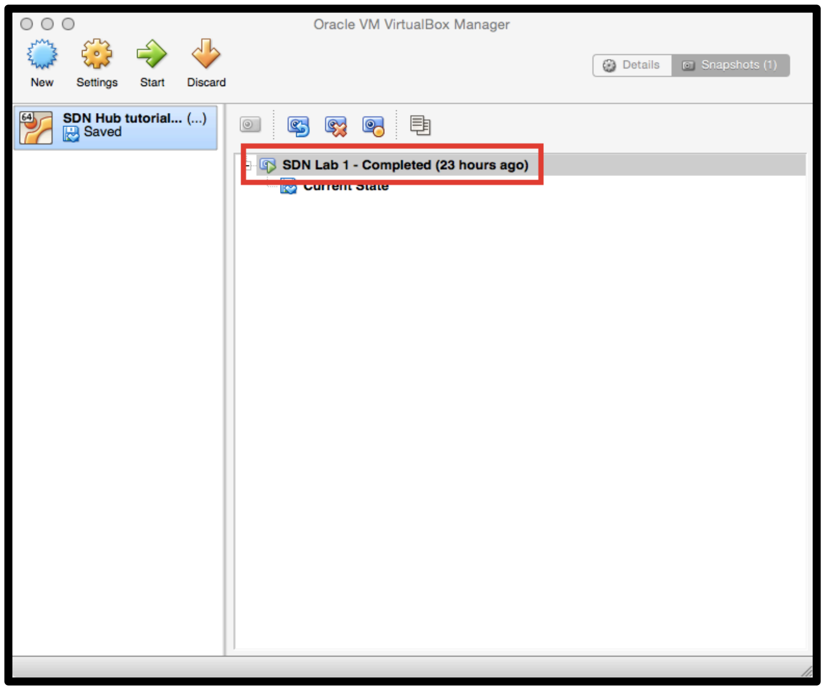
Network

USB
Device Filters: 0 (0 active)

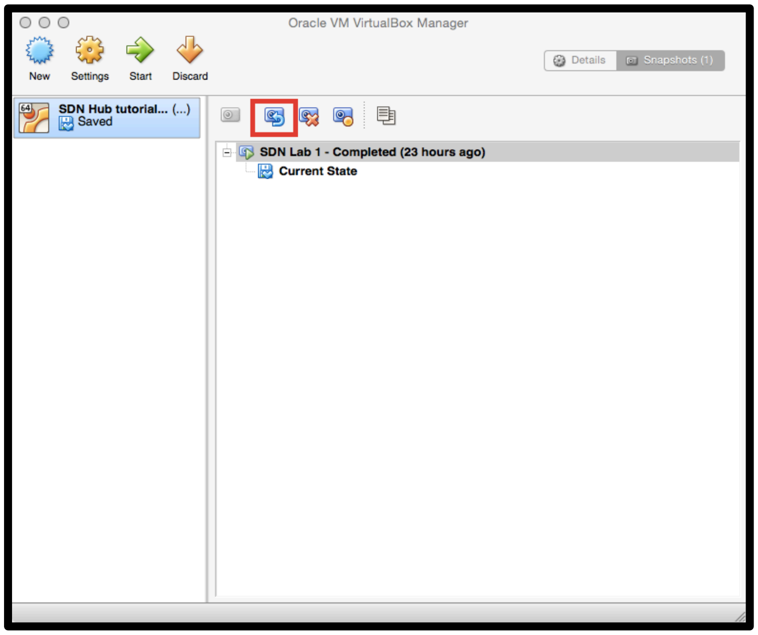
Shared folders
None

Description
Ubuntu 14.04 64-bit

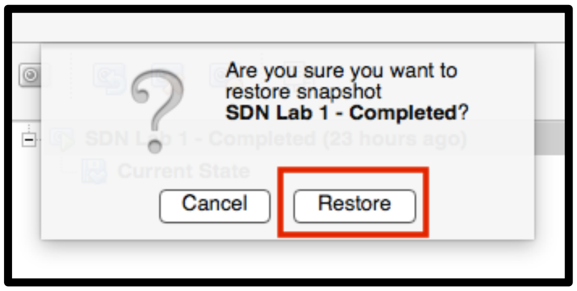
Step 2: Within your VirtualBox application window, select the snapshot state that you want to recover:


Oracle VM VirtualBox Manager
New
Settings
Start
Discard
Details
Snapshots (1)
SDN Hub tutorial... (...)
Saved
SDN Lab 1 - Completed (23 hours ago)
Current State

Step 3: Within your VirtualBox application window, select the “Restore Snapshot” icon:


Oracle VM VirtualBox Manager
New
Settings
Start
Discard
Details
Snapshots (1)
SDN Hub tutorial... (...)
Saved
SDN Lab 1 - Completed (23 hours ago)
Current State

Step 4: You will now be greeted with a confirmation dialog. Click the “Restore” button:


Are you sure you want to restore snapshot **SDN Lab 1 - Completed**?

Congratulations!

Your saved state has been restored, and you are ready to start your virtual machine.