Software Defined Networking (SDN)
Lab 1: OpenFlow Communications

ACI-REF
Advanced CyberInfrastructure –
Research and Education Facilitators

Version 1.1
**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](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/](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:
ACI-REF:  OpenFlow Lab Guide

b. You should now see the VirtualBox application window:

![VirtualBox Application Window]

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

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

![VirtualBox Import Appliance]

---

5
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.

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”.

e. From the resulting window, click “Import”. For the purpose of this lab exercise, we will accept the default virtual machine settings.
**Step 3:** Launch the virtual machine.

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

![VirtualBox Manager](image1.png)

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

![VirtualBox Manager](image2.png)
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).

**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 capable 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](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.

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

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

and press “Enter”. Your terminal should look like this:
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 terminal output](image)

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
**ACI-REF: OpenFlow Lab Guide**

**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
   ```

![Terminal window showing cd command output]

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

   ```
mvn install -nsu
   ```

![Terminal window showing mvn install command output]
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"):

```
```

![Terminal window showing OpenDaylight directory](image)

```
Ubuntu@sonnhubvm:~/SDNHub_OpenDaylight_Tutorial/distribution/opendaylight-osgi-adsal/target/distribution-osgi-adsal-1.1.0-SNAPSHOT-osgipackage/opendaylight[14:36 ] (master)$
```

![Terminal window showing OpenDaylight process](image)

```
2015-05-21 14:39:17.177 PDT [Start Level Event Dispatcher] INFO o.o.c.c.i.ConfigurationService - ConfigurationService Manager init
2015-05-21 14:39:39.904 PDT [SwitchHandler-1] INFO o.o.c.p.o.core.internal.Content - Switch: 127.0.0.1:330767 is connected to the Controller
2015-05-21 14:39:39.905 PDT [SwitchHandler-2] INFO o.o.c.p.o.core.internal.Content - Switch: 127.0.0.1:330764 is connected to the Controller
2015-05-21 14:39:39.910 PDT [SwitchHandler-3] INFO o.o.c.p.o.core.internal.Content - Switch: 127.0.0.1:330765 is connected to the Controller
2015-05-21 14:39:40.067 PDT [Start Level Event Dispatcher] WARN o.o.c.u.internal.i.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.TutorialInstallL2Forwarding - Initialized
2015-05-21 14:39:40.198 PDT [Start Level Event Dispatcher] INFO o.o.t.t.i.TutorialInstallL2Forwarding - Started
```

```
osgi> ss tutorial
"Framework is launched."
```

```
id 55
bundle org.sdnhub.odel.tutorial.adsal.L2_forwarding_0.5.0-SNAPSHOT
```

```
State ACTIVE
```

![Terminal window showing ss tutorial command](image)

![Terminal window showing framework status](image)

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.
**ACI-REF: OpenFlow Lab Guide**

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

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

![Printnodes output](image)

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’).

![Logical topology](image)

**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:

![Terminal Emulator window with pingall command output]

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:
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:
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:

```
State ACTIVE
```

```
osgi> ss tutorial
"Framework is launched."
```

```
id  State         Bundle
ss   ACTIVE       org.sdnhub.odl.tutorial.adsal_L2_forwarding:0.5.0-SNAPSHOT
```

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](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:
**ACI-REF: OpenFlow Lab Guide**

**Step 3:** Within the “Capture” column of the Wireshark application, select the “**Loopback: lo**” interface:

![Capture Options](image)

**Step 4:** Press the “**Start**” button:

![Start](image)

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

Step 2: Enter a snapshot name and description (optional) into the resulting dialog box. Note that your virtual machine is now paused:
**Step 3:** Select the “**OK**” button. You will now receive a status window that depicts the progress of the snapshot:

![Snapshot Status Window](image)

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:

![VirtualBox Snapshot Button]

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

![VirtualBox Snapshots List]
**Step 3:** Within your VirtualBox application window, select the “Restore Snapshot” icon:

![VirtualBox Manager](image)

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

![Confirmation Dialog](image)

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