**CELL iteration 2: 4 Objectives 09/05/16**

1. Start VM, login (mininet/mininet) and obtain ip (**sudo dhclient eth1**)
2. Run Xming
3. Putty to VM; run wireshark then filter of ; capture->interfaces->loopback (**sudo wireshare &**)
4. Putty to VM run ODL (**cd ~/opendaylight; sudo ./run.sh -virt ovsdb**)
5. Browser to host open IP of VM (admin/admin) [**http://192.168.56.101:8080**](http://192.168.56.101:8080)

**Objective 2.1 Learn about OpenDaylight Controller and SDN**

**Task 2.1.Open Daylight sites and Videos**

OVS = Open virtual Stack

ODL 127.0.0.1:8080 will use port 6633

ODL <http://192.168.56.101:8080>

**Objective 2.2.1 Get OpenDaylight running with a mininet test network**

**Task 2.2 Run OpenDaylight controller; perform exercises with Mininet**

**mininet; Initial setup: Putty to VM start mininet**

**sudo mn --controller=remote,ip=127.0.0.1,port=6633**

**ODL: Browser to host open IP of VM (admin/admin) <http://192.168.56.101:8080>**

nodes appear in Nodes Learned, simple topology (refresh page if needed)

**iteration2.2.1.A.InitialSetupDefault.png**

**ODL:** tried the more complex topology

**sudo mn --controller=remote,ip=127.0.0.1,port=6633 --topo tree,3**

**iteration2.2.1.B.InitialSetupMoreComplex.png**

**ODL: 1to2:** Create openflow on the switch and install the flows

**iteration2.2.1.C.InstallFirstFlow.1to2.Dialog.png**

**iteration2.2.1.D.InstallFirstFlow.1to2.png**

**ODL: 2to1:** Create reverse openflow on the switch

**iteration2.2.1.E.InstallSecondFlow.2to1.png**

**mininet: pingall**

**Verify: sudo ovs-vsctl show**

**iteration2.2.1.F.verify.png**

**Pingall: end up with 7 nodes under Devices, 15 flows under Flows**

**iteration2.2.1.G.pingall.png**

**Objective 2.3: Run Simple Forwarding NetApp see how ODL ctrl makes forwarding decisions**

Task 2.3 Hands on Training using Simple Forwarding Net Apps, **Run ODL Simple Forwarding NetApp**

**mininet; Initial setup: Putty to VM start mininet**

**sudo mn --controller=remote,ip=127.0.0.1,port=6633 --topo tree,3**

I noted that mininet unable to connect to controller, and it has **8 hosts and 7 switches**

**iteration2.3.1.H.mininet.Topo.png**

**iteration2.3.1.I.ODL.Topo.png**

**Add the gateway to the controller**

**iteration2.3.1.K.AddGateway.png**

**iteration2.3.1.L.AddGateway.png**

**Pingall: All hosts are now reachable**

1. The 8 hosts will be **discovered using ARP messages**. Address Resolution Protocol is used by Layer 2 to resolve ip address to MAC addresses and to maintain cache table of MAC to ip addresses. ARP is the way by which any host on a LAN can dynamically learn the MAC of another host on the same **lan**. The **ARP request**, is a **broadcast**, goes like this, if this is your ip address, please reply back with your MAC address.
2. Destination only/32 entries are installed across all switches along with the corresponding output ports toward that host. This is my first few attempts, here I am only seeing 2 flow details under the troubleshooting tab

**mininet:** success seen in mininet

**iteration2.3.1.M.mininet.pingall.png**

**eration2.3.1.N.pingall.verify.png**

**ODL** on troubleshoot tab, display flow details; This was showing me OF|1 and OF|2

**At this time I was not seeing flow details for all hosts**

**iteration2.3.1.O1.pingall.png (same as picture G, this was on switch 1)**

**iteration2.3.1.O2.ODL.pingall.switch2.png (this one was on switch 2)**

\*\*\*\* this is the part I had trouble with in getting the output on the troubleshooting flow details \*\*\*\* to match what was presented in the instructions. When I followed these steps, then I was \*\*\*\* able to see 12 entries in the troubleshooting siwtch1 flows in ODL browser

1. Start ODL: **cd opendaylight; ./run/sh –virt ovsdb**
2. I did a clear of mininet 3 times: **sudo mn –c; sudo mn-c’ sudo mn –c**
3. Run mininet: **sudo mn --controller=remote,ip=127.0.0.1,port=6633 --topo tree,3**
4. Open the browser:[**http://192.168.56.101:8080**](http://192.168.56.101:8080)
5. Uninstall all **NORMAL** default flows for hosts, but I left the NORMAL host for switch1
6. Make sure the flows for 1to2 and 2to1 are installed
7. pingall

**iteration2.3.1.P1.topo.png**

**iteration2.3.1.P2.DeviceTab.png**

**iteration2.3.1.P3.FlowTab.png**

**iteration2.3.1.P4.TroubleshootingTab.png**

**on OSGI console** (command line interface on window where started the ODL on VM)

**iteration2.3.1.Q.ODL.osgi.png**

In ODL on troubleshoot tab, load flow details of one of the switches, I selected switch 6

On the Flows tab, I noted all the flows turned green.

Additional Notes:

<http://archive.openflow.org/wk/index.php/OpenFlow_Tutorial>

**Objective 2.4 Run, analyze and interpret Learning Switch and Simple Firewall NetApps**

**2.4.1.1.1 HUB: Run Python Ethernet Learning Switch NetApp**

1. Started the mininet, and noted it was unable to connect to the controller

mininet@mininet-vm:

sudo mn --topo single,3 --mac --switch ovsk --controller remote

\*\*\* Creating network

\*\*\* Adding controller

**Unable to contact** the remote controller at **127.0.0.1:6633**, controller is at local host port 6633

2. Started the mininet, and noted it was unable to connect to the controller, and checked ip

h1-eth0 Link encap:Ethernet HWaddr 00:00:00:00:00:01

pingall failed as expected because the switch has nothing in its flow table

Machines are not logically connected

**iteration2.4.1.R.HubsetupPingFail.png**

3. additional notes found here

<http://studylib.net/doc/9508117/mininet-and-openflow-labs>

4. commands, dump the content of the flow table of the OpenFlow Switch

**dpctl show tcp:127.0.0.1:6634**

**dpctl dump-flows tcp:127.0.0.1:6634**

mininet@mininet-vm:~$ dpctl dump-flows tcp:127.0.0.1:6634

stats\_reply (xid=0x2e59e5c8): flags=none type=1(flow)

**FLOW TABLE IS EMPTY**

5. start the POX controller with **HUB**; create a controller running on local machine port 6633 to connect with the openflow switches (adds listener for OpenFlow, floods the packets on all ports on the switch ) pox/forwarding/hub.py)

**iteration2.4.1.S.POXcontrollerHUB.png**

**OPENFLOW switches listen on port 6634**

**now pingall succeeds and dump flow shows data (ETHERNET HUB)**

mininet@mininet-vm:~$ dpctl **dump-flows** tcp:127.0.01:6634

stats\_reply (xid=0x794d776c): flags=none type=1(flow)

cookie=0, duration\_sec=346s, duration\_nsec=320000000s, table\_id=0, priority=32768, n\_packets=24, n\_bytes=1680, idle\_timeout=0,hard\_timeout=0,actions=**FLOOD**

**2.4.1.1.2 LEARNING SWITCH; Run Python Learning Switch NetApp**

Enhance it to make it an intelligent Ethernet Switch, stores MAC to determine output port.

Learning.py implements the switch

**iteration2.4.1.T.POXlearningSwitch.png**

**2.4.1.2 Simple Firewall: Run Python Simple Firewall NetApp**

Extend the learning switch to make decisions based on firewall rules installed at the POX openflow controller. Drop the packet if there is a matching firewall entry set to false, or no entry at all. Only forward the packet if firewall entry is found and set to true.

Our test, only macs :01 and :03 will be processed and forwarded; h2 packets are dropped

**/home/mininet/freshpox/pox/pox/forwarding/simple\_firewall.py**

Copied it from: <https://raw.githubusercontent.com/shankarchari/mininet_mysteries/master/firewall/srcs-l2_firewall.py> and then I changed this line from self.AddRule('00-00-00-00-00-01',EthAddr('00:00:00:00:00:02')) to self.AddRule('00-00-00-00-00-01',EthAddr('00:00:00:00:00:03'))

**iteration2.4.1.U.simplefirewall.pingall.png**

**iteration2.4.1.V.simplefirewall.h1pingh3.png**