4 Objectives 8/22/16

**Objective 1.1: learn fundamentals of SDN**

**iteration1.1.A.SDN.jpg**

**iteration1.1.B.wireshark.png**

**SDN:**

Architectural framework, separation of forwarding from control plane

3 layers, Application (orchestration) Layer, SDN control Layer, Infrastructure Network function Data Layer

API (northbound) Control Plane (southbound) Data Plane

Well defined open standards based protocol for northbound/soutbound

openflow – protocol for interacting with behavior of switches/routers

**CONTROL PLANE**

Create control planes on common hardware, centralized locations

The openflow conference controller is a network computer

Remotely access network equipment using openflow

**DATA PLANE**

Layer 2 OSI model switches, transfer data within a local network

Layer 3 OSI model gateway between different networks

**Wireshark**

Graphical utility for viewing packets; Filtered on “of” (openflow)

**Objective 1.2: Access and Learn mininet**

**MININET:**

open source network emulator

running hosts, switches, controllers in a single linux kernel created using software

You can create different and custom topologies

Switches and controllers in mininet are in root name space (top of DNS hierarchy .com, .net)

Hosts in mininet are in own network namespace

Controller/switches/hosts are run in our mininet VM in Virtual Box

Controller acts as an Ethernet Learning switch with the Openflow Switch

**CLI Basic Commands**

**iteration1.2.A.basicCommands.png**

**Run regression tests self-contained/different topologies**

(sudo mn – test <>) (Examples of tests you can run; pingpair, iperf)

**iteration1.2.B.regressionTests.png**

**Parameterized topology (sudo mn –test pingall –topo single,3)**

**iteration1.2.C.changingTopology.png**

**Run python interpreter**

**iteration1.2.D.pythonInterpreter.png**

**Run python API examples**

**iteration1.2.E.pythonExamplesDir.png**

**Objective 1.3: Linux, Python in Python NetApps**

1. sudo mn --custom topo-2sw-2host.py --topo mytopo
2. sudo python topox.py

**Run custom mininets**

**iteration1.3.A.CustomMininet.png**

**topox.py script**

**iteration1.3.B.topox.png**

**Run python API examples (miniedit, gui for designing the topology)**

**iteration1.3.C.MiniEdit.png**

Running things in the examples directory (Examples):

<http://www.brianlinkletter.com/how-to-use-miniedit-mininets-graphical-user-interface/>

mininet@mininet-vm:~/mininet/examples$ sudo ./miniedit.py

**Objective 1.4: run, analyze, and interpret python applications using Pox OpenFlow Controller**

**POX – CONTROLLER (pythonic OpenFlow) (remote controller running outside the VM)**

After we setup the network environment, we need to install entries to the switch so packets can be forwarded among the hosts. Pox is an openflow controller, lets you spin up controllers.

**Initialsetup**

/home/mininet/freshpox/pox

**iteration1.4.A.InitialSetup.png**

**iteration1.4.B.hub.floodInterfaces.png**

**Ping a non existent host, get 3 unanswered ARP requests**

**iteration1.4.C.hub.pingfail.png**

**iteration1.4.C.hub.pingfail2.png**

**Change to a hub**

/home/mininet/freshpox/pox/pox/misc

**iteration1.4.D.hub2switch.png**

**iteration1.4.D.switch.png**

sudo killall controller

<https://github.com/mininet/openflow-tutorial/wiki/Create-a-Learning-Switch>

**NOTES:**

dpctl command-line utility that sends basic OpenFlow messages, useful for viewing switch port and flow stats, plus manually inserting flow entries