Installing and Using Mininet and Wireshark

(No submission)

1): Installing VirtualBox

The first thing required for Mininet is a Virtual Machine (VM) manager to run our Mininet VM. VirtualBox is free and open source and can be downloaded here. You are free to use kvm, vmware, or xen if you are familiar with those hypervisors - however, the TAs will not support them, so you are on your own.

If you are having trouble installing VirtualBox, make sure to consult the VirtualBox manual, Piazza, or the TAs. If you do not have a computer, please contact the TAs and we can work something out.

Common Problems Installing VirtualBox:

- Did you download the correct version? (32-bit vs 64-bit)
- Is your computer old? It might not be able to be virtualized -- talk to the TA.
- Make sure that virtualization is enabled in your BIOS.

2): Installing Mininet and Wireshark

Once VirtualBox is installed, we can install the Mininet VM. There are different ways to install the Mininet VM. For this class, we provide a guide to first install ubuntu 20.04 on the VirtualBox then install Mininet and Wireshark software. Detailed instruction is as follows:

Ubuntu 20.04 virtual machine installation

1. Download ubuntu20.04 desktop iso file from https://releases.ubuntu.com/focal/ .
2. Click on “New” on VirtualBox GUI, type a name (e.g., Ubuntu 2004) and select your downloaded .iso file in the “ISO Image” box, then type your username and password, keep “Next” till “Finish” with the default settings.
3. Double click and power up your Ubuntu virtual machine.

Wireshark installation:

1. Boot the Ubuntu VM and finish the OS installation.
    a. Note that if you are using VirtualBox 7+ and chose **unattended** installation, your user will not be added to sudoers! Use the following commands to fix it. Reboot or log out when you are finished.
        i. su –
        ii. sudo adduser ‘your username’ sudo
2. Open the terminal (ctrl+alt+t or press the super key and search terminal) and type
    a. sudo apt update
    b. sudo apt install wireshark
        i. Choose “yes” in the prompted window, it is asking you to give the current user root privileges to use Wireshark
    c. Then you can use `sudo wireshark` in terminal to open Wireshark

Mininet installation on Ubuntu 20.04:

Compile from source (Option 2 in http://mininet.org/download/).

a. git clone https://github.com/mininet/mininet
b. git checkout -b mininet-2.3.0
c. cd ..
d. mininet/util/install.sh

Part III: Practice - Using Mininet

A walkthrough can be found on the Mininet page. First and foremost, here's some background and information on what Mininet is and why we are using it.

What is Mininet? Mininet is a "network in a box" developed at Stanford in 2010. It is software designed to allow large scale networks to be emulated in software on a laptop. Its rise has also been dictated by the use of OpenFlow (which will be the subject of Lab 4 and the Final Project). If you are interested in reading, here is the original Mininet paper.

Why Mininet? Mininet is probably one of the simplest forms of network emulators, is *free*, is open source, and is widely used by the research community, as well as by universities for teaching computer networks. It allows for more interesting topologies than what can be achieved in the physical lab.

How does Mininet Work? Mininet works simply by creating a virtual network on your computer/laptop. It accomplishes this task by creating host namespaces (h1, h2, etc) and connecting them through virtual interfaces. So, when we run ping between the linux namespaces h1 and h2, the ping will run from h1's namespace through a virtual interface pair created for h1 and h2, before it reaches h2. If h1 and h2 are connected through a switch as shown in the python code in the Mininet walkthrough, the ping will transit multiple virtual interface pairs. The switches that we will be using are running OpenVSwitch (OVS). Mininet will connect additional virtual interfaces between each virtual port on the switch with each connected host. The host name space allows each host to see the same file system but operates as its own process that will run separately from each of the other host processes. The OVS version running on the Ubuntu image supports OpenFlow.

Understanding some mininet commands:

1. *sudo mn*: will start mininet

```
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>
```

2. You can type *help* after you have run mininet net.

```
mininet> help

Documented commands (type help <topic>):
========================================
EOF    gterm  iperfudp  nodes        pingpair      py      switch
dpctl  help   link      noecho       pingpairfull  quit    time
dump   intfs  links     pingall      ports         sh      x
exit   iperf  net       pingallfull  px            source  xterm

You may also send a command to a node using:
  <node> command {args}
For example:
  mininet> h1 ifconfig

The interpreter automatically substitutes IP addresses
for node names when a node is the first arg, so commands
like
  mininet> h2 ping h3
should work.

Some character-oriented interactive commands require
noecho:
  mininet> noecho h2 vi foo.py
However, starting up an xterm/gterm is generally better:
  mininet> xterm h2
```

3. We can see that when we launched mininet, it created a mini-network from the output. When we use the *net* command we can see *h1* indicated host 1, it has one network interface *eth0* which is connected to the switch on interface *eth1*. This is shown on the output line: *h1 h1-eth0:s1-eth1*. There is also host 2 (*h2*), switch 1 (*s1*), and controller 0 (*c0*).

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet> nodes
available nodes are:
c0 h1 h2 s1
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=1049>
<Host h2: h2-eth0:10.0.0.2 pid=1052>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=1057>
<Controller c0: 127.0.0.1:6633 pid=1042>
```

   a. From dump we can also see what IP address have been assigned to h1 and h2. Also, pids are given to each process. Processes in mininet are used for hosts, switches, and controllers. Mininet is composed of processes using Interprocess Communication (IPC) to emulate a network environment.

Running Mininet as a Python script:

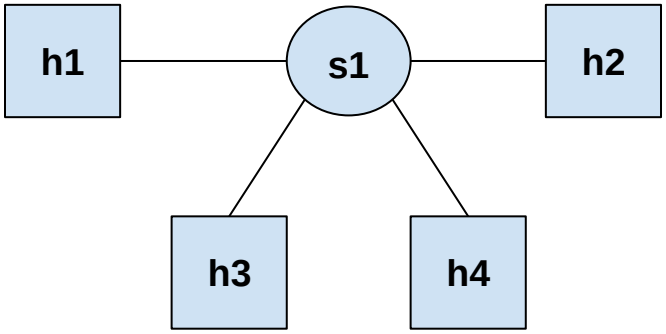
To make custom topologies, it is useful to be able to refine a topology in a script. The following is an example of using a Python script to launch Mininet with a custom topology:

```
#!/usr/bin/python

from mininet.topo import Topo
from mininet.net import Mininet
from mininet.cli import CLI

class MyTopology(Topo):
    """
    A basic topology
    """
    def __init__(self):
        Topo.__init__(self)

        # Set Up Topology Here
        switch = self.addSwitch('s1')    ## Adds a Switch

        host1 = self.addHost('h1')       ## Adds a Host

        self.addLink(host1, switch)      ## Add a link

if __name__ == '__main__':
    """
    If this script is run as an executable (by chmod +x), this is
    what it will do
    """

    topo = MyTopology()           ## Creates the topology
    net = Mininet( topo=topo )          ## Loads the topology
    net.start()                      ## Starts Mininet

    # Commands here will run on the simulated topology
    CLI(net)

    net.stop()                       ## Stops Mininet
```

You could use this as a skeleton to practice using Mininet. On the Mininet site, The API Reference will be an excellent resource for figuring out how to run pings or open the command prompt in between the net.start() and net.stop() lines.

Practice Problem:

1. In Mininet change the default configuration to have 4 hosts connected to a switch.



2. Save a screenshot of *dump* and *pingall* output. Explain what is being shown in the screenshot.
3. Run the *iperf* command as well, and screenshot the output, how fast is the connect?
4. Run wireshark, and using the display filter, filter for “openflow_v1”. Note: When you run wireshark you should do so as “sudo wireshark”. When you choose an interface to capture on, you should select “any”.
    a. Run ping from a host to any other host using *hX ping -c 5 hY*. Show the screenshot on the Mininet terminal. And take a screenshot to show the captured related packets by Wireshark.
    b. Find packets that matches the “openflow_v1” filter with the OpenFlow typefield set to OFPT_PACKET_IN and *OFPT_PACKET_OUT respectively*. What is the source and destination IP address for these two entries? Answer the question and take screenshots showing your results.
    c. Replace the display filter to “icmp and not openflow_v1”. Run *pingall* again, how many entries are generated in wireshark? Show the number and take a screenshot of your results.