CEG5101 Modern Computer Networking

Graduate Assistant: Mr. Binghui Wu and Ms. Divya D Kulkarni,
Prepared by: Ms. Anum Talpur (Ex-Graduate Assistant)
Lecturer: Assoc. Prof. Mohan Gurusamy
ECE Department, CDE., National University of Singapore

Laboratory 01

Introduction to Mininet

LEARNING OBJECTIVES

Upon successful completion of this laboratory, the students will be able to learn:

i. To download and install Mininet.
ii. To become familiar with the Mininet Interface.

EQUIPMENT

i. *Hardware* – Laptop/Computer with internet access
ii. *Software* – Mininet, Linux OS

DISCUSSION

Task 1: To download and install Mininet

1.1 Mininet is a network emulator for prototyping a large network on a single machine.

1.2 It is a virtual environment that allows network administrators to experiment and run a collection of network devices, such as routers, switches, and other related network devices on a Linux kernel.

1.3 Mininet is used to design simple as well as complex custom-defined topologies.

1.4 Mininet offers a user-friendly environment for development and research and allows users to deploy custom networks through the CLI and/or GUI interfaces.

1.5 Mininet is an open-source project, and its source code can be explored on https://github.com/mininet

1.6 To download and install Mininet, you need a Linux environment. NOTE: If you are not using Linux, follow and complete lab-00 first before proceeding to the next steps.

1.7 To download and get started with Mininet, proceed to http://mininet.org/download/ and choose "Option 2: Native Installation from Source" as shown below.

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/

Download/Get Started With Mininet

The easiest way to get started is to **download a pre-packaged Mininet/Ubuntu VM**. This VM includes Mininet itself, all OpenFlow binaries and tools pre-installed, and tweaks to the kernel configuration to support larger Mininet networks.

- Option 1: Mininet VM Installation (easy, recommended)
- Option 2: Native Installation from Source
- Option 3: Installation from Packages
- Option 4. Upgrading an *existing* Mininet Installation
- **Important Note**: Python 2 and Python 3 Mininet

1.8 Next, open the terminal on Ubuntu and follow the given steps to complete the installation process.

1.9 The first step is to get the source code using the command, "`git clone https://github.com/mininet/mininet`"

```
ubuntu@sdnlab:~$ git clone https://github.com/mininet/mininet
```

```
Cloning into 'mininet'...
remote: Enumerating objects: 10211, done.
remote: Counting objects: 100% (57/57), done.
remote: Compressing objects: 100% (40/40), done.
remote: Total 10211 (delta 23), reused 40 (delta 16), pack-reused 10154
Receiving objects: 100% (10211/10211), 3.23 MiB | 3.74 MiB/s, done.
Resolving deltas: 100% (6805/6805), done.
```

1.10 Next, open the mininet folder (using the command "`cd mininet`") and check for the latest available version using the command "`git tag # list available versions`". You can continue with any version, but it's recommended to install the latest one.

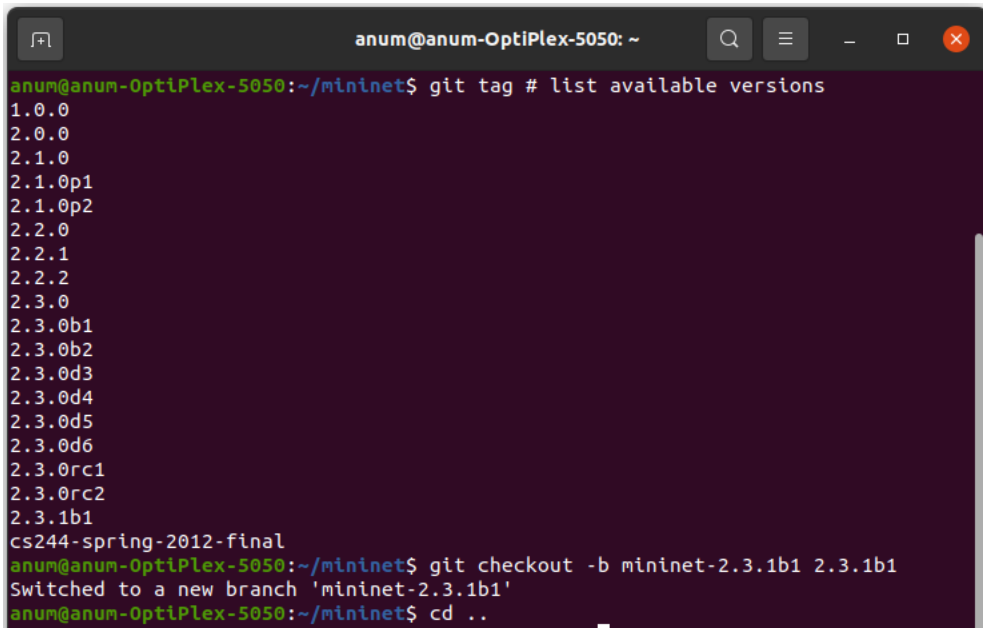
```
ubuntu@sdnlab: ~/mininet
ubuntu@sdnlab:~$ cd mininet
ubuntu@sdnlab:~/mininet$ git tag # list available versions
1.0.0
2.0.0
2.1.0
2.1.0p1
2.1.0p2
2.2.0
2.2.1
2.2.2
2.3.0
2.3.0b1
2.3.0b2
2.3.0d3
2.3.0d4
2.3.0d5
2.3.0d6
2.3.0rc1
2.3.0rc2
2.3.1b2
2.3.1b3
2.3.1b4
cs244-spring-2012-final
ubuntu@sdnlab:~/mininet$
```

1.11 Choose your version for installation with the command "`git checkout -b mininet-2.3.1b4 2.3.1b4`". Note: I have chosen the latest available version here.

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/

```
ubuntu@sdnlab:~/mininet$ git checkout -b mininet-2.3.1b4 2.3.1b4
Switched to a new branch 'mininet-2.3.1b4'
ubuntu@sdnlab:~/mininet$
```

1.12 Next, go back to the source tree or home location of your PC with the command "`cd ..`".



1.13 Once you are at the root location, then type the command "`mininet/util/install.sh [options]`" to begin the installation process.

```
To install everything (using your home directory): install.sh -a
To install everything (using another directory for build): install.sh -s mydir -a
To install Mininet + user switch + OvS (using your home dir): install.sh -nfv
To install Mininet + user switch + OvS (using another dir:) install.sh -s mydir -nfv
```

For example, I installed everything: "`mininet/util/install.sh -a`"

```
ubuntu@sdnlab:~$ mininet/util/install.sh -a
Detected Linux distribution: Ubuntu 22.04 jammy amd64
sys.version_info(major=3, minor=10, micro=12, releaselevel='final', serial=0)
Detected Python (python3) version 3
Installing all packages except for -eix (doxypy, ivs, nox-classic)...
Install Mininet-compatible kernel if necessary
[sudo] password for ubuntu:
Hit:1 http://cn.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://cn.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://cn.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:5 https://ppa.launchpadcontent.net/deadsnakes/ppa/ubuntu jammy InRelease
Reading package lists... Done
Reading package lists...
Building dependency tree...
```

1.14 Once the installation is complete, you will see the process ended with a message "Enjoy Mininet!".

```
Enjoy Mininet!
```

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/

1.15 To test the mininet, type "`sudo mn`" and you will see a small network of 2 hosts connected with a single switch and a controller is created, and the mininet is initiated as shown below.

```
ubuntu@sdnlab:~$ 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>
```

1.16 This confirms the successful installation of the Mininet.

1.17 Then type "exit" to shut down the Mininet

```
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 51.162 seconds
ubuntu@sdnlab:~$
```

1.18 It is optional but recommended to sign up for the mininet-discuss mailing list at https://mailman.stanford.edu/mailman/listinfo/mininet-discuss to get support and discuss with the Mininet community.

Task 2: To become familiar with the Mininet Interface

2.1 In this activity, we will become familiar with the Mininet environment and start with building a simple network by connecting two end devices with the switch.

2.2 After completing the installation, launch a terminal and type "`sudo mn -h`" to get help describing Mininet options and commands. This gives a list of commands to be used to create a network emulation.

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/

```
ubuntu@sdnlab:~$ sudo mn -h
Usage: mn [options]
(type mn -h for details)

The mn utility creates Mininet network from the command line. It can create
parametrized topologies, invoke the Mininet CLI, and run tests.

Options:
  -h, --help            show this help message and exit
  --switch=SWITCH       default|ivs|lxbr|ovs|ovsbr|ovsk|user[,param=value...]
                        user=UserSwitch ovs=OVSSwitch ovsbr=OVSBridge
                        ovsk=OVSSwitch ivs=IVSSwitch lxbr=LinuxBridge
                        default=OVSSwitch
  --host=HOST           cfs|proc|rt[,param=value...] proc=Host
                        rt=CPULimitedHost{'sched': 'rt'}
                        cfs=CPULimitedHost{'sched': 'cfs'}
  --controller=CONTROLLER
                        default|none|nox|ovsc|ref|remote|ryu[,param=value...]
                        ref=Controller ovsc=OVSController nox=NOX
                        remote=RemoteController ryu=Ryu
                        default=DefaultController none=NullController
  --link=LINK           default|ovs|tc|tcu[,param=value...] default=Link
                        tc=TCLink tcu=TCULink ovs=OVSLink
  --topo=TOPO           linear|minimal|reversed|single|torus|tree[,param=value
                        ...] minimal=MinimalTopo linear=LinearTopo
                        reversed=SingleSwitchReversedTopo
                        single=SingleSwitchTopo tree=TreeTopo torus=TorusTopo
  -c, --clean           clean and exit
  --custom=CUSTOM       read custom classes or params from .py file(s)
  --test=TEST           pingall|pingpair|iperf|iperfudp|all|none|build
  -x, --xterms          spawn xterms for each node
  -i IPBASE, --ipbase=IPBASE
                        base IP address for hosts
  --mac                 automatically set host MACs
  --arp                 set all-pairs ARP entries
  -v VERBOSITY, --verbosity=VERBOSITY
                        debug|info|output|warning|warn|error|critical
  --innamespace         sw and ctrl in namespace?
  --listenport=LISTENPORT
                        base port for passive switch listening
  --nolistenport        don't use passive listening port
```

2.3 Following the above help, the command to build a custom topology of a single switch and two hosts is "`sudo mn --topo single,2`"

```
ubuntu@sdnlab:~$ sudo mn --topo single,2
*** 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>
```

2.4 The default topology is a minimal topology, and it can be changed with the command "`--topo=NAME`". The available NAME for defining different topologies can be found from the output of "`sudo mn -h`" in step 2.2.

2.5 To find out more about the current network topology, type "`net`" and enter. The window is shown below. It shows that port **eth0** of host **h1** is connected to port **eth1**

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/

of switch **s1.** Similarly**,** port **eth0** of host **h2** is connected to **eth2** of a switch **s1**. Also, switch **s1** has a loopback interface **lo**.

```
*** Starting CLI:
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet>
```

2.6 Mininet allows users to issue a specific command on a device with the format of "DEVICE-NAME COMMAND". The first string typed into Mininet CLI is the name of the host, controller, or switch, and the command typed next to the device name is the one executed on that node. As an example, to check the configured IP address for the host h1, type "`h1 ifconfig`", and for host h2 type "`h2 ifconfig`". To print the process list from host h1 type "`h1 ps -a`" and so on.

```
mininet> h1 ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.1  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::4cdf:27ff:fea7:e890  prefixlen 64  scopeid 0x20<link>
        ether 4e:df:27:a7:e8:90  txqueuelen 1000  (Ethernet)
        RX packets 38  bytes 4004 (4.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 11  bytes 866 (866.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

mininet>
```

2.7 To find and explore more on available Mininet commands, type "`help`". As an example, the command "`nodes`" display nodes, "`links`" displays links and their active status, and so on.

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

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
```

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/

2.8 You can check network connectivity by sending a ping command from one host to another. (**control + C** to quit the ping process)

```
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=1.21 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.177 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.119 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.044 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.047 ms
```

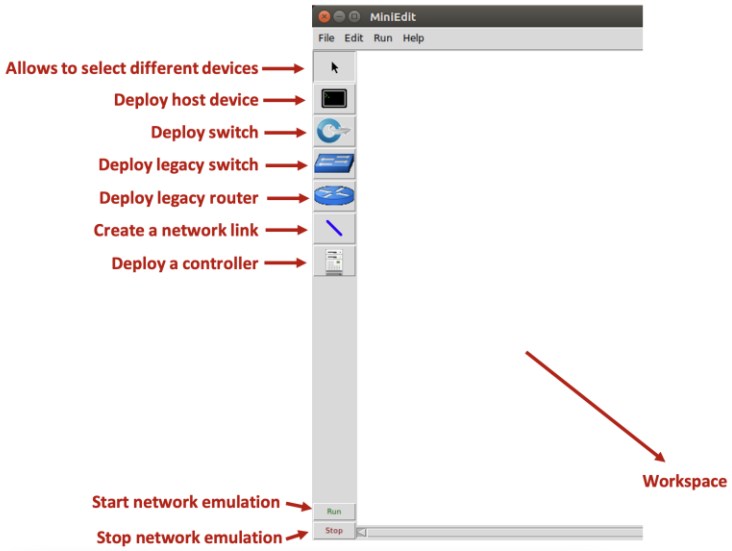
2.9 To leave Mininet and end the current network setup, type "`exit`".

```
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 354.358 seconds
```

2.10 Next, we will learn to use the GUI interface of Mininet, known as MiniEdit.

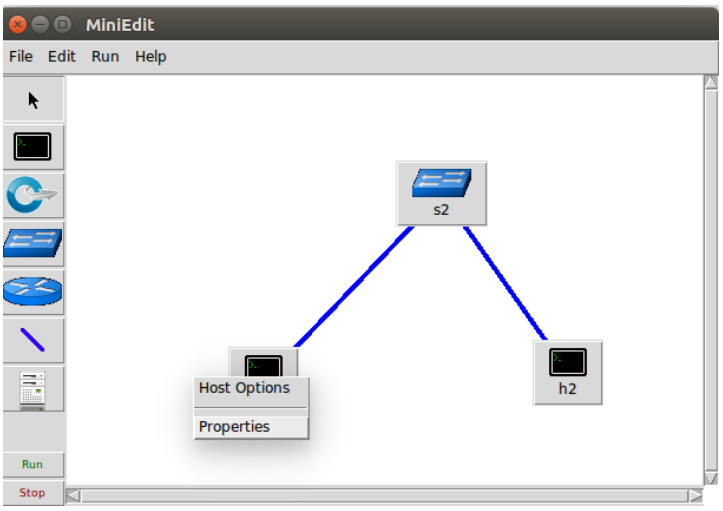
2.11 To get started with MiniEdit, locate the folder where mininet is installed and type "`sudo python3 mininet/examples/miniedit.py`" The default interface window for MiniEdit is shown below. (if you are using python2, you can use the command : "`sudo ~/mininet/examples/miniedit.py`")

2.12 You can move your cursor around different icons and a help option will appear with an explanatory title. Some of the key interface bars and their functioning is highlighted below.



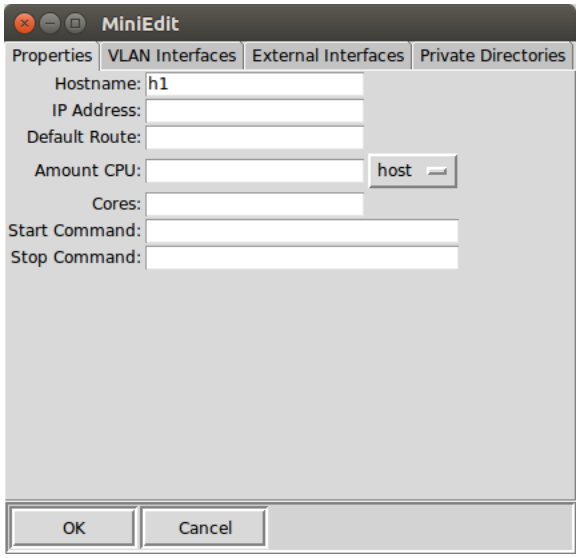
2.13 We can add devices to the workspace by selecting the device type from the sidebar and clicking it on the workspace. A link between two devices can be created by dragging the link option from one device and dropping it on another device.

2.14 Next, to configure a device, right-click on the icon, and select the `properties`.

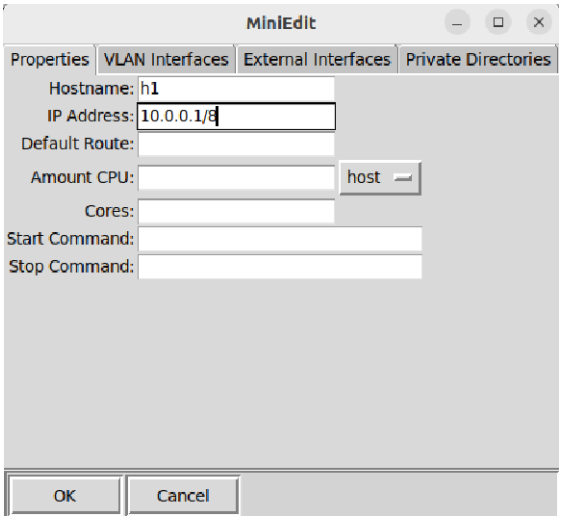


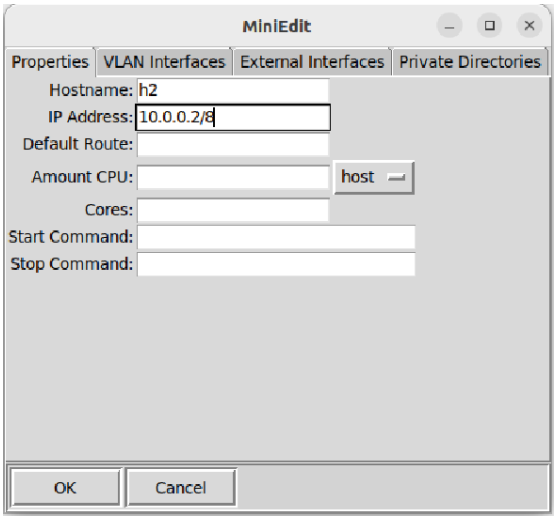
2.15 Below shows the property window of host h1 and switch s1, to add or remove different parameters to the devices.

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/



2.16 We can configure the IP address of host h1 and host h2 using the device property window. As an example, we use 10.0.0.1/8 for host h1 and 10.0.0.2/8 for host h2.

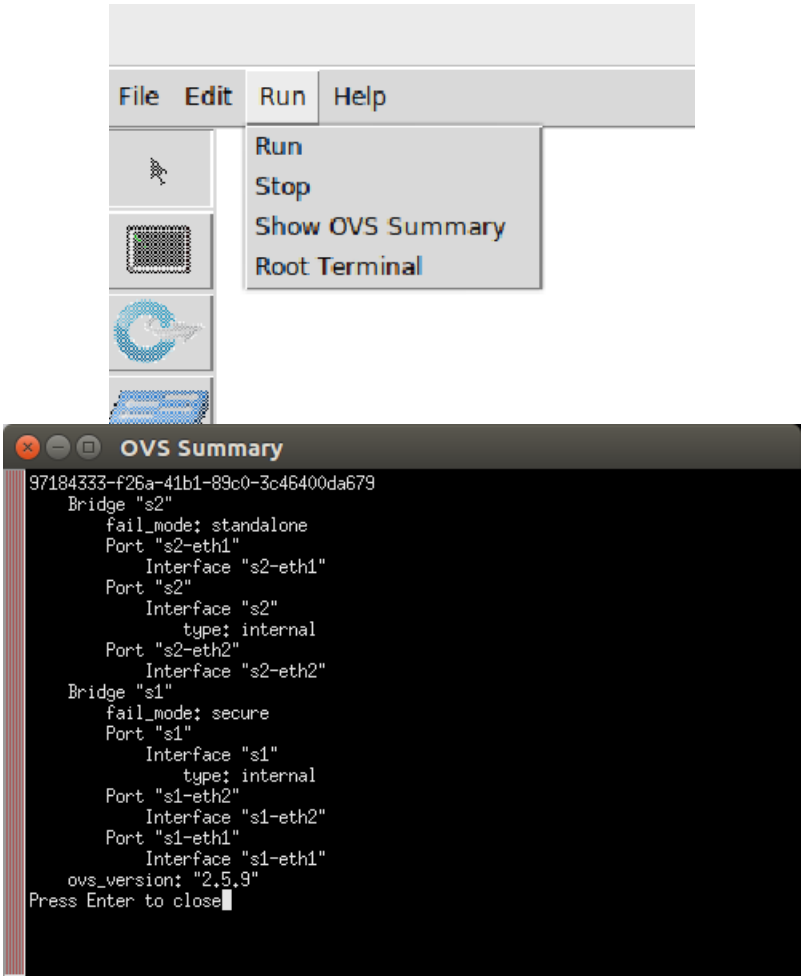




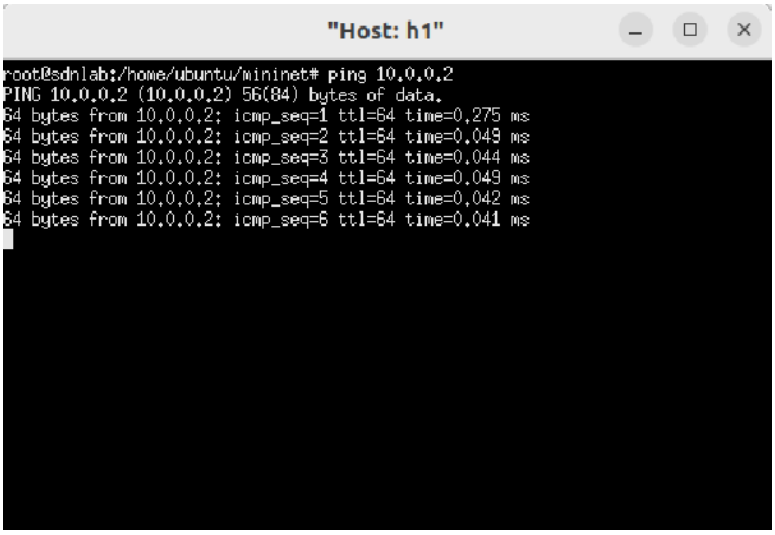
2.17 To start the simulation, click on the "`Run`" button.

2.18 After starting the simulation, we can view the network summary by clicking "`Show OVS summary`" from the `Run` tab.

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/



2.19 We can also check connectivity between devices by sending a `ping` command from one host to another. To do so, right-click on the host and select `terminal`. Type `ping` from one host with the address of another host.



2.20 Finally, to save your file as a completed network and for future use, click on `File>save` and then save it with `.mn` (mininet activity file) file format.

https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
http://mininet.org/walkthrough/