CAN304 Lab 5

Virtual Machine and Mininet (Virtual Network)

In this lab, you will learn the virtual machine (which we have mentioned before but that is not necessary for the cryptographic labs) and the virtual network (i.e., mininet), which will be used as the testbed for the network attack and defense experiments later on.

1. Virtual Machine

1.1. Why do we need virtual machine (VM)?

- A physical machine can run multiple VMs (guest OSes)
- Run Linux guest OS (if your host OS is Windows)
- Much safer doing network security lab against VM
- SDN lab-friendly

1.2. How would you get VM hypervisor?

- VirtualBox (open-source)(recommended) [1]
- VMware (not free)(you may find cracked version for free usage) [2]
- Other VM hypervisors: QEMU, KVM, UML, etc.

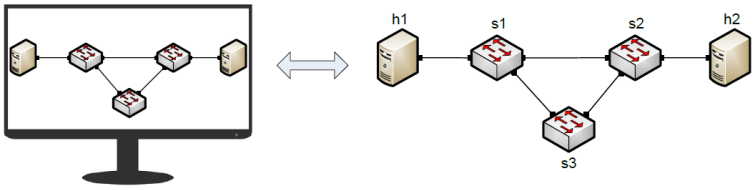
1.3. How would you create a VM?

- First, you need to find a guest operating system (OS) image file. Ubuntu (version 18.04 or later) is recommended. E.g., Ubuntu 20.04 LTS [3]
- Once you download the Ubuntu OS image, you can start to create the VM [4][5]:
  - Use the VM hypervisor, e.g.,Virtualbox, to create a VM box.
  - Install the Ubuntu OS image on the VM box.

2. Virtual network – Mininet

1.1 Introduction

- Mininet is a virtual testbed enabling the development and testing of network tools and protocols. With a single command, Mininet can create a realistic virtual network on any type of machine (Virtual Machine (VM), cloud-hosted, or native). Therefore, it provides an inexpensive solution and streamlined development running in line with production networks.



1.2 Mininet offers the following features:

- Fast prototyping for new networking protocols.
- Simplified testing for complex topologies without the need of buying expensive hardware.
- Realistic execution as it runs real code on the Unix and Linux kernels.
- Open source environment backed by a large community contributing extensive documentation.

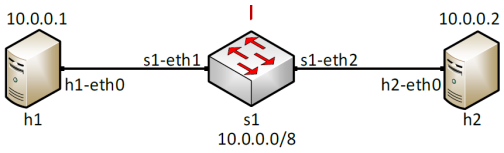
1.3 Practice with Mininet

- Install Mininet on Ubuntu:
  - $ sudo apt-get install mininet
- Test if it is installed successfully
  - $ sudo mn

```
*** No default OpenFlow controller found for default switch!
*** Falling back to OVS Bridge
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller

*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

- The default minimal network topology



- Command line interface: mininet>
- Command “help” shows the list of Mininet CLI commands and examples on their usage: mininet>help

```
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

mininet>
```

- To display the available nodes, use the following command: mininet> nodes

```
mininet> nodes
available nodes are:
h1 h2 s1
```

- To display the links to understand the topology, use the command: mininet> net

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
```

The output of this command shows that:
1) Host h1 is connected using its network interface *h1-eth0* to the switch on interface *s1-eth1*.
2) Host h2 is connected using its network interface *h2-eth0* to the switch on interface *s1-eth2*.
3) Switch s1:
   a. has a loopback interface *lo*.
   b. connects to *h1-eth0* through interface *s1-eth1*.
   c. connects to *h2-eth0* through interface *s1-eth2*.

- Mininet allows you to execute commands on a specific device. To issue a command for a specific node, you must specify the device first, followed by the command: mininet> h1 ifconfig

```
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::94f4:a9ff:febe:93cc  prefixlen 64  scopeid 0x20<link>
        ether 96:f4:a9:be:93:cc  txqueuelen 1000  (Ethernet)
        RX packets 47  bytes 4820 (4.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 15  bytes 1146 (1.1 KB)
        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
```

This command executes the ifconfig Linux command on host h1. The command shows host h1's interfaces. The display indicates that host h1 has an interface *h1-eth0* configured with IP address 10.0.0.1, and another interface lo configured with IP address 127.0.0.1 (loopback interface).

Homework:

1. Install VM using Ubuntu guest OS and install Mininet on the Ubuntu VM
2. Show and describe the display after executing this command: h1 ping 10.0.0.2
3. Try to use another mininet command (except for "exit") and show the display

Reference

[1] https://www.virtualbox.org/wiki/Downloads

[2] https://www.vmware.com/products/workstation-player.html

[3] https://ubuntu.com/download/desktop

[4] https://www.ktexperts.com/how-to-install-ubuntu-20-04-1-lts-on-windows-using-virtualbox/

[5] https://www.youtube.com/watch?v=x5MhydijWmc