**If your computer is Windows, or is Apple MacBook with Intel Chip, follow Part A, and skip Part B.**

**If your computer is Apple MacBook with M1/M2/M3/M4…, skip Part A, and follow Part B.**

**Part A.**

**Installing and Using Mininet For Windows**

## Installing Virtualbox

The first thing required to use the Mininet VM is to install a Virtual Machine manager – software that allows for multiple “virtual machines” to be run on a single “machine”. The manager we recommend is VirtualBox – it is *free*, open source and can be downloaded [here](https://www.virtualbox.org/wiki/Downloads). You are free to use kvm, Vmware, or Xen if you are familiar with those hypervisors - however, the TAs will not support them, so if you choose this route you are on your own for support.

Installation Problems:

Are you having trouble installing VirtualBox? Make sure to consult the VirtualBox [manual](https://www.virtualbox.org/manual/ch02.html) and/or ask questions during the lab. It is recommended that you **start the lab early** in case you encounter any problems setting up either VirtualBox or Mininet.

Common Problems Installing VirtualBox:

* Did you download the appropriate package for your platform? (32-bit vs 64-bit)
* Did you download the correct version? (7.0 is recommended )
* Is your computer really old? It might not be able to use virtualization -- talk to the TA.
* Make sure that virtualization is enabled in your BIOS.

If VirtualBox fails with "Error: Fatal error during installation":

* To see details about the error, use the command: "C:\path\to\virtualBoxFile.exe" --logging --msi-log-file "C:\path\to\outputFile.txt"
* If outputFile.txt has "ERROR\_NO\_CONFIGMGR\_SERVICES" as the reason for failing, install Microsoft Configuration Manager at "[https://www.microsoft.com/en-us/evalcenter/  
  download-microsoft-endpoint-configuration-manager](https://www.microsoft.com/en-us/evalcenter/download-microsoft-endpoint-configuration-manager)".
* Run the downloaded Microsoft Configuration Manager exe file.
* On the same directory, find "cd.latest\SMSSETUP\Tools\SupportCenter\  
  SupportCenterInstaller.msi" and run "SupportCenterInstaller.msi".
* Use the "Windows + R" keyboard shortcut and enter "services.msc" into the window.
* Check the status of "SMS Agent Host" as running and reinstall VirtualBox

If there are no errors during setup but the VM doesn’t work properly when launched, just try rebooting.

## Installing Mininet

Once VirtualBox is installed, you have two options to set up and manage your VM. You can do so directly from *Virtualbox* or you can use *Vagrant* which provides a command-line version.

For clear instructions on what to do next, please refer to the steps below depending on your chosen option.

**Import OVA file into Virtualbox:**

1. Download the Mininet OVA file [here](https://drive.google.com/file/d/1rh7r4zyrztgZFmVL4d8SWGXcxmaAjCC0/view?usp=drive_link). (You must use your UCSC Google account to get access.)
2. Once the download is complete, open VirtualBox and select File>Import Appliance… Navigate to the Mininet OVA file you downloaded.
3. When the Mininet VM has been successfully imported, start the VM.
4. You should be presented with a GUI.

Helpful tips:

1. Chromium is a web browser. You can use this to go to the class webpage and copy/paste example code from the PDFs.
2. You can use VirtualBox shared folders to copy files to/from the VM. The TAs or student tutor can help you with this in the lab.

**Part B.**

**Installing and Using Mininet VM for Apple MAC M1/M2**

## Using UTM

##### Download and install UTM (free): ​​<https://github.com/utmapp/UTM/releases/tag/v4.4.5>

##### Download the Mininet qocw2 file [here](https://drive.google.com/file/d/1d3hVL5VjZJDbzSUYs0OlH43zYj5oGx1u/view?usp=share_link). (You must use your UCSC Google account to get access.)

Import with UTM

Follow the [instructions](https://docs.google.com/presentation/d/1V9uRjvbFIdJGoZn39lrVFUHhazAul1rPhs8Mn0Uh4DE/edit#slide=id.p)

## Extend Reading:

<https://harshdadhich.com/blog/linux-vm/>

**Part C.**

**Frequently Asked Questions (FAQ)**

1. **How do I move files from my personal computer to the VM?**

In Virtualbox there are a few ways you can transfer files, however, it is known to be tricky. Here is a [link](https://www.makeuseof.com/tag/transfer-files-virtual-machine-guest-host-pc/) for sharing files. Note: If you are using Vagrant, check out [this](https://developer.hashicorp.com/vagrant/docs/synced-folders/basic_usage) tutorial for sync setup.

1. **VM out of storage – if your VM runs out of space:**

While working with the provided Mininet VM during this quarter, you might run into an issue where you can no longer save your code or any files due to lack of space.

Thankfully, in order to free up a lot of space, you can simply run the following command:  
 sudo rm -rf "/home/mininet/VirtualBox VMs"

Explanation: The folder shown above contains a lot of files that are not needed and are quite large (around 2.6 GB in VM images), and the command will fully remove them. This won't give you any issues and should free up more than enough space for the whole quarter. You can do this preemptively to avoid the lack of space issue or simply do it after you've found out you've run out of space.

Additionally, if you ever run into trouble where you have files backed up only inside the VM, but due to some issue can no longer boot up the VM, the files on the VM's disk are still accessible, you just have to go through a process to retrieve them.

1. **Traceroute times out when running in the VM in the UCSC PC lab:**

Follow the instructions in Lab 2 to run the VM in BRIDGED mode.

Linux Prelab

This pre-lab is designed to get you comfortable with a Linux environment. These commands can all be run on any Linux machine you have access to. You can [SSH to the campus timeshare](http://its.ucsc.edu/unix-timeshare/tutorials/how-to-connect.html) if you do not have a Linux machine easily available. You can also set up the Mininet VM early (see Lab 1 in this PDF) and use that as a Linux environment. Within the VM, the Terminal application will provide you with a CLI.

**Recommended resources for this lab:**

<http://linux-training.be/index.php?nav=fundamentals>

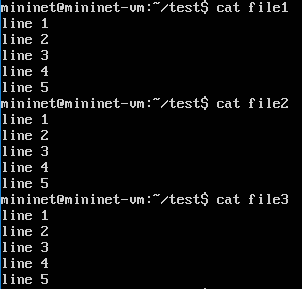
<http://linuxcommand.org/lc3_writing_shell_scripts.php>

Feel free to also use online resources/Google/StackOverflow/etc. to help you find solutions to these questions. If you use an online resource, make sure to give credit. Providing the URL in your answer PDF is adequate.

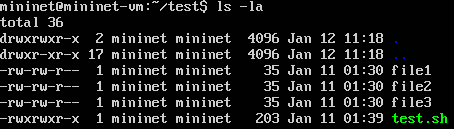
## Pre-lab Questions:

1. [5] What command will show you which groups you are a member of?
2. [5] What does the environmental variable “$?” hold? (Hint: the command ‘echo $?’ will show you this on your screen)
3. [5] What key combination will suspend a currently running process and place it as a background process?
4. [5] With what command (and arguments) can you find out your kernel version and the “nodename”? [The output should not include any other information]
5. [5] What is the difference between the paths “.”, “..”, and “~”? What does the path “/” refer to when not preceded by anything?
6. [5] What is a pid? Which command would you use to find the “pid” for a running process?
7. [20] Write a single command that will return every user’s default shell. [You may chain commands using piping and redirects] (Hint: See ‘Chapter 19: filters’ of linux-training.be as well as the man page for the /etc/passwd file: <https://linux.die.net/man/5/passwd>)
8. [10] What is the difference between “sudo” and “su root”?
9. [10] How would you tell your computer to run a program or script on a schedule or set interval on Linux? E.g. Run this program once every 30 minutes.
10. [30] Write a shell script that only prints the even numbered lines of each file in the current directory. The output should be *filename: line* for each even numbered line. You do not need to print line numbers.

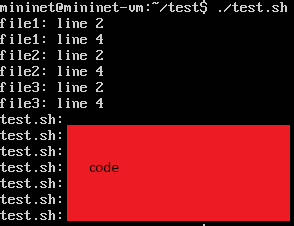
For example, let’s say I write a script in a file called test.sh. I then create 3 files, as shown in the screenshot below:



My directory now looks like the following:



When I run the script, the output should look like the following.



## Using Mininet

A walkthrough can be found on the mininet [page](http://mininet.org/walkthrough/#part-1-everyday-mininet-usage). First and foremost, here’s some background and information on what is Mininet and why we are using it over the physical lab in BE301A.

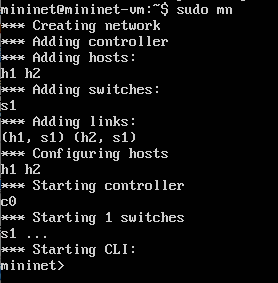
**What is Mininet?** Mininet is a “network in a box” developed at Stanford in 2010. It is software designed to allow large-scale networks to be emulated in software on a laptop. Its rise has also been dictated by the use of OpenFlow (which will be the subject of Lab 4 and the Final Project). If you are interested in reading, here is the original Mininet [paper](http://conferences.sigcomm.org/hotnets/2010/papers/a19-lantz.pdf).

**Why Mininet?** Our physical lab is aging from the wear and tear of 11 years of usage (it was donated by Cisco in 2004). 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](http://conferences.sigcomm.org/sigcomm/2014/doc/slides/mininet-stanford-experiences.pdf) computer networks. It allows for more interesting topologies than what can be achieved in the physical lab. Most importantly, it has enabled us to accommodate CE 150/L’s enrollment growth (only five years ago enrollment was around 40 students, while today it is peaking at 100). The physical lab only has 10 workstations, however, certain workstations are lacking parts (which either have not been or will be replaced) leaving even fewer usable stations per lab. So in light of the growing student base, we are looking towards a solution that scales with the number of students. Almost all students have access to a computer, so instead of buying hundreds of thousands of dollars worth of specialized equipment (e.g., routers, switches), we use general-purpose computers (as a side note: if you don’t have a personal computer please talk to the TAs about using VirtualBox on the lab computers).

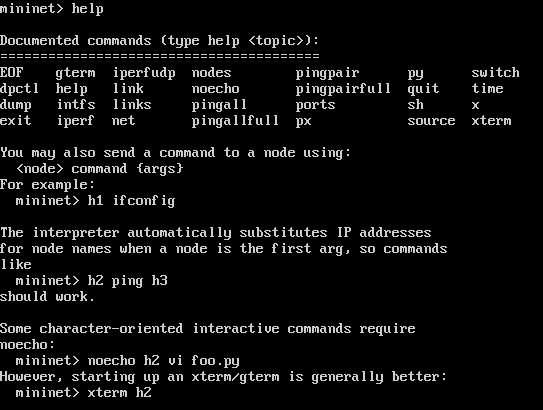
**How does Mininet Work?** Mininet works simply by creating a virtual network on your computer/laptop. It accomplishes this task by creating host namespaces (h1, h2, etc) and connecting them through virtual interfaces. So when we run ping between the Linux namespaces h1 and h2, the ping will run from 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:

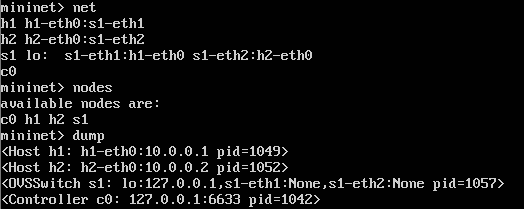
1. *sudo mn*: will start mininet



1. You can use *mn -h* or type *help* after you have run mininet net.



1. 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*).



* 1. From dump we can also see what IP address have been assigned to h1 and h2. We also are given the pids of 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. You can review the prelab’s resources and reading to understand more.

### 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 should use this as a skeleton for getting started on the lab. On the Mininet site, The [API Reference](http://mininet.org/api/index.html) 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.

### The Lab [100 pts]:

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



1. [30 pts] Save a screenshot of *dump* and *pingall* output. Explain what is being shown in the screenshot.
2. [10 pts] Run the *iperf* command as well, and screenshot the output, how fast is the connect?
3. Run wireshark, and using the [display filter](https://www.wireshark.org/docs/wsug_html_chunked/ChWorkDisplayFilterSection.html), typing “openflow\_v1” in filter line and remember to click *Apply* after you start. Note: When you run wireshark you should do so as “sudo wireshark” in a new terminal. When you choose an interface to capture on, you should select “any”.
   1. [20 pts] Run ping from a host to any other host using *hX ping -c 5 hY*. How many *of\_packet\_in* messages show up? Take a screenshot of your results.
   2. [20 pts] What is the source and destination IP addresses for these entries? Find another packet that matches the “of” filter with the OpenFlow typefield set to *OF\_PACKET\_OUT*. What is the source and destination IP address for this entry? Take screenshots showing your results.
   3. [20 pts] Replace the display filter for “of” to “icmp && not of”. Run *pingall* again, how many entries are generated in wireshark? What types of icmp entries show up? Take a screenshot of your results.

**Submission:**

You will submit 4 files for this assignment:

1. <your name>-lab1.pdf
   1. The PDF with all of your solutions to the Prelab and Lab.
2. <your name>-script.sh
   1. The shell script you created in Prelab 1. This should be marked as executable.
3. <your name>-topo.py
   1. The mininet topology you created in Lab 1.
4. README
   1. A README file. This should describe the contents of each file you submit, and contain your name, e-mail, CruzID, and student ID number.