LAB: DoH vs non-DoH Packet Capture

1. Some Import Information Before You Start

The Domain Name System: Simplifying Networking:

The Domain Name System (DNS) emerged in the early 1980s to streamline networking processes. During the late 1960s and 1970s, the initial networks, like ARPANET, comprised a small number of hosts—only four nodes in ARPANET's case. However, as the 1980s approached, the surge in network users made remembering IP addresses a cumbersome task.

Before the creation of DNS, a straightforward text file named "HOSTS.txt" was used to map all existing sites. With the mounting need for a more efficient system to simplify networking, the concept of DNS was born.

DNS essentially associates names with various internet resources and stores this information in servers known as domain name servers. Every time you access a new website, you are essentially communicating with one of these servers, even though it may go unnoticed. This ingenious system enables users to input user-friendly names like "amazon.com" instead of the more intricate IP addresses such as "127.0.0.1"

Moreover, DNS servers not only store IP addresses but also house other crucial pieces of information, such as Start of Authority (SOA) records, mail exchanges, name servers, and more. Each piece of information is intricately linked to a specific name.

Hypertext Transfer Protocol (HTTP): A Cornerstone of the World Wide Web

Hypertext Transfer Protocol (HTTP) came into existence during the early 1990s, marking a pivotal moment in networking history, especially with the advent of the World Wide Web. HTTP was designed to facilitate the exchange of information over the internet, essentially becoming the backbone of web communication.

The inaugural version of HTTP was incredibly simplistic by today's standards. It primarily utilized GET commands to retrieve information from external servers, with the response typically delivered in a plain text format. During this phase, HTTP lacked the extensive elements we now consider standard, such as HTTP headers, footers, status codes, and error codes.

Fast forward to the present, and HTTP has become the cornerstone of internet communication. Its applications have diversified, encompassing information retrieval, data exchange, and even real-time updates of web pages. The ubiquity of HTTP within the internet landscape firmly establishes its enduring presence, ensuring its continued prominence as long as the internet persists.

Domain Name Resolution with HTTP (DoH): Enhancing Privacy and Security

Domain Name over HTTP (DoH) represents a modern approach to conducting DNS lookups. Unlike traditional DNS lookups that employ the DNS protocol, DoH utilizes HTTP for these queries. This shift not only streamlines the process but also introduces a significant privacy advantage. By integrating DNS functionality into the HTTP protocol, DoH makes it exceptionally challenging for potential attackers or eavesdroppers to intercept and interpret the transmitted information.
The heightened privacy is a direct result of HTTP's encryption of data between the DNS client and the DNS resolver. This encryption renders the transmitted data indecipherable to anyone attempting to capture and inspect network contents.

Introduced in 2017, DoH has experienced remarkable growth. Many popular web browsers have swiftly incorporated support for DoH, underscoring its increasing adoption and acknowledgment within the digital landscape.

Transmission Control Protocol (TCP): A Fundamental Networking Protocol:

The Transmission Control Protocol (TCP) stands as one of the most pervasive networking protocols in use today. Initially conceived in the 1970s, TCP was designed to work in tandem with the Internet Protocol (IP), which eventually became the foundational protocol of the modern internet. In the early 1980s, another significant IP protocol, the User Datagram Protocol (UDP), was introduced and gained widespread use.

TCP distinguishes itself from UDP—the User Datagram Protocol—by being a connection-oriented protocol. In this context, a connection is established only when both parties involved initiate it. The connection initiation process follows a three-way handshake: the host wishing to commence the connection sends a synchronization (SYN) message to the intended host, which responds with a synchronized acknowledgment (SYN/ACK) message. This exchange signifies their mutual intent to establish a connection. Subsequently, the initial host acknowledges (ACK) the message, completing the handshake, and enabling the exchange of messages between the two hosts.

Moreover, TCP incorporates error correction mechanisms, unlike UDP. If a packet is received in a damaged state or does not arrive at all, the receiving host refrains from sending an acknowledgment (ACK) message. Consequently, the sending host retransmits the packet, ensuring data integrity. However, this inherent error correction capability comes at the cost of speed, making TCP relatively slower than IP. As a result, TCP is primarily utilized for data transmission scenarios where a slight delay in the order of seconds is imperceptible, such as in email exchanges or large file downloads. Conversely, for real-time applications like video conferencing or live TV streaming, where minimizing lag is crucial, other protocols like UDP are favored over TCP.

Understanding TCPdump: Capturing and Analyzing Network Traffic:

TCPdump is a command-line utility that allows you to capture and analyze network traffic flowing into and out of your system. This tool is a valuable asset for network administrators, aiding in troubleshooting network-related issues by providing insights into the traffic exchanged by a device.

When you run the TCPdump command, it displays information about the traffic entering and exiting your device. This information can be presented directly on the screen or saved into a file. The utility offers several flags to customize the data capture, allowing you to specify what data to capture, how many packets to capture, and whether to write it into a file.

In addition to flags, TCPdump supports filter expressions, enabling you to collect specific data based on predefined criteria. Here are some common flags and filter expressions to enhance your usage:
- -i eth0: Captures packets from the eth0 interface.
- -c <number of packets>: Specifies the number of packets to capture.
- -w <filename>: Writes the captured packets into a file.
- tcp: Captures only TCP packets.
- -v: Increases the verbosity level of the output, providing more detailed information for each packet. You can use it up to 3 times (e.g., -vvv) for even more detailed output.

2. Pre Lab-Questions

1. What does the Domain Name System (DNS) do?

2. What is the name of the servers the Domain Name System stores the domain names for Ip addresses?

3. What are some of the differences between the current version of HTTP and some of the earlier versions?

4. How does DNS over HTTP (DoH) improve on traditional DNS and what advantages does it bring in terms of privacy and security?

5. How has DNS over HTTP (DoH) improved since it was first released in 2017. Do some research.

6. Give some scenarios in which TCP would be preferred over UDP and vice versa. Think about their different functions and tradeoffs like speed, reliability, etc…

7. What are some possible scenarios where TCP dump is useful in terms of network administration.

3. Laboratory Instructions: Capturing and Analyzing DoH and non-DoH Packets

Overview:

Now that you have gained an understanding of DNS, HTTP, DoH, and TCP dumps, let's proceed to the lab. Here, you'll capture packets into a .pcap file and analyze them using the doHlyzer tool, which will generate a .csv file for further analysis.

Typically, this lab doesn't demand sudo privileges. However, if the need arises, you can acquire them using the password: '***password123***'.

Lab Installation

This lab starts from the Labtainer working directory on your Linux host in your virtual machine (VM). If you've previously run labs on your VM, you can skip the following step. However, for a brand new VM without any prior lab activity, execute the command:

mkdir /home/student/.local/share/labtainers[1]

This command creates a folder used to store the link necessary to retrieve your lab.

Next, navigate to your 'labtainer-student' directory and use the following command to download the lab onto your VM:

imodule github.com/ML4CYB/DoH_Labtainer_tar_file/raw/main/imodule.tar

Once the download is complete, update the scripts by running:

update-labtainer.sh[2]

[1] This command serves the sole purpose of creating a directory to store the web address used for obtaining the .tar file that initializes the lab.

[2] Ensure you execute this command, as it updates all necessary files and libraries for downloading additional data required by the lab. Failure to run this script before starting the lab may result in a crash.

To begin the lab, enter the command:

labtainer doh-lab[3]

This command will connect the resulting virtual terminal to a client computer.

This may take a few minutes to install and initialize.

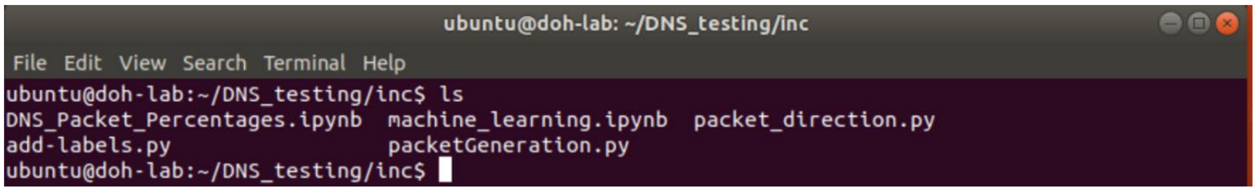
Tasks

a. **Navigate to the *DNS_testing/inc* directory.**

Use the change directory command "cd" to navigate to the DNS_testing/inc directory.

b. **Take a look at your directory.**[4]

Use the **ls** command to list the contents of the directory. You'll notice several files, including:


ubuntu@doh-lab: ~/DNS_testing/inc
File Edit View Search Terminal Help

```
ubuntu@doh-lab:~/DNS_testing/inc$ ls
DNS_Packet_Percentages.ipynb  machine_learning.ipynb  packet_direction.py
add-labels.py                 packetGeneration.py
ubuntu@doh-lab:~/DNS_testing/inc$
```

- **packetGeneration.py**: This script generates internet traffic.
- **add-labels.py**: This script appends labels (doH or non-doH) to each packet.
-**machine_learning.ipynb** – jupyter notebook script used for machine learning aspect of the lab
-**DNS_Packet_Percentages.ipynb** – jupyter notebook script used in the networking aspect of the lab.

c. **Update dependencies.**

Run **sudo apt-get update** to make sure all dependencies are up to date.

[3] When running this command for the first time, all the scripts and libraries required for the lab will be extracted from a Docker repository on Docker Hub. Please be aware that the process may take some time as it involves downloading and extracting a great amount of data.

[4] In the directory, you will see an additional scripts: packet_direction.py. It is used to initialize the DoHlyzer program, but it's important to note that they are not required for use in the lab.

d. Download Necessary Libraries and Software

In this section, you will acquire several essential libraries required for this lab.

1. Download tcpdump:

Execute the following command to obtain tcpdump, a software tool that will be used to capture network traffic:

```
sudo apt-get install tcpdump
```

This will install tcpdump, which will capture the packets being produced by the packetGeneration.py script.

2. Download Google Chrome:

While the graphic interface won't be utilized in this lab, Google Chrome will run in the background alongside chromedriver for performing lookups and generating packets. Execute the following three commands:

```
wget https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb
```

```
sudo dpkg -i google-chrome-stable_current_amd64.deb
```

```
sudo apt-get install -f
```
[5]

[5] This command is employed for the installation of Chrome. The -f flag is utilized to address potential broken dependencies that may arise during the Chrome installation process. You might still encounter some dependency issues while installing Chrome; however, these can be disregarded. It's important to note that, due to the Labtainer infrastructure, the graphical interface of Chrome may not function optimally. Nevertheless, the backend, which is the part you will be utilizing, works seamlessly.

These commands fetch the Google Chrome installation file, download it into your lab environment, repackage the file, and install it onto your system.

3. Install Selenium

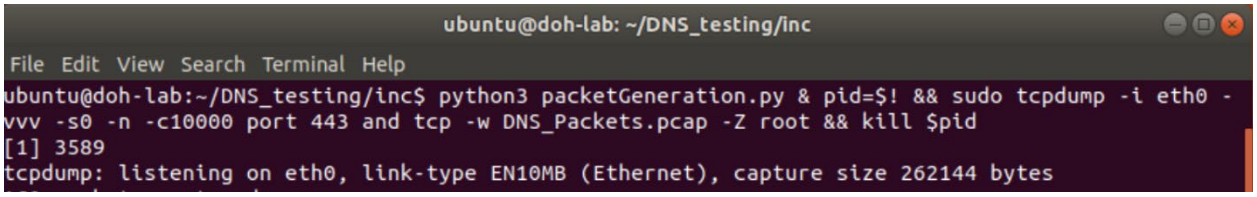
The Selenium library is a powerful tool for automating web browsers. For our use it will enable the Python script to perform GET requests and conduct website lookups. To install execute the following command:

pip install selenium

Ensure you follow these steps to set up the necessary components for the upcoming lab exercises.

e. **Execute the TCP dump command to capture packets and run the Python script in the background using the & command.**

python3 packetGeneration.py & pid=$! && sudo tcpdump -i eth0 -vvv -s0 -n -c10000 port 443 and tcp -w DNS_Packets.pcap -Z root && kill $pid[6]


ubuntu@doh-lab: ~/DNS_testing/inc
File Edit View Search Terminal Help

```
ubuntu@doh-lab:~/DNS_testing/inc$ python3 packetGeneration.py & pid=$! && sudo tcpdump -i eth0 -
vvv -s0 -n -c10000 port 443 and tcp -w DNS_Packets.pcap -Z root && kill $pid
[1] 3589
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Note the flags being used, such as -i for interface, -w for writing to a .pcap file, and -n for capturing packets on port 443.

f. **Once done run ls again to see a new file.**

You will notice a new file

DNS_Packets.pcap

[6] The command is designed to terminate the execution of the packetGeneration.py script once tcpdump captures the specified number of packets, in this case, 10,000. The use of the -Z root option indicates that the command is executed with root user privileges, enabling it to generate and store data in the DNS_Packets.pcap file.

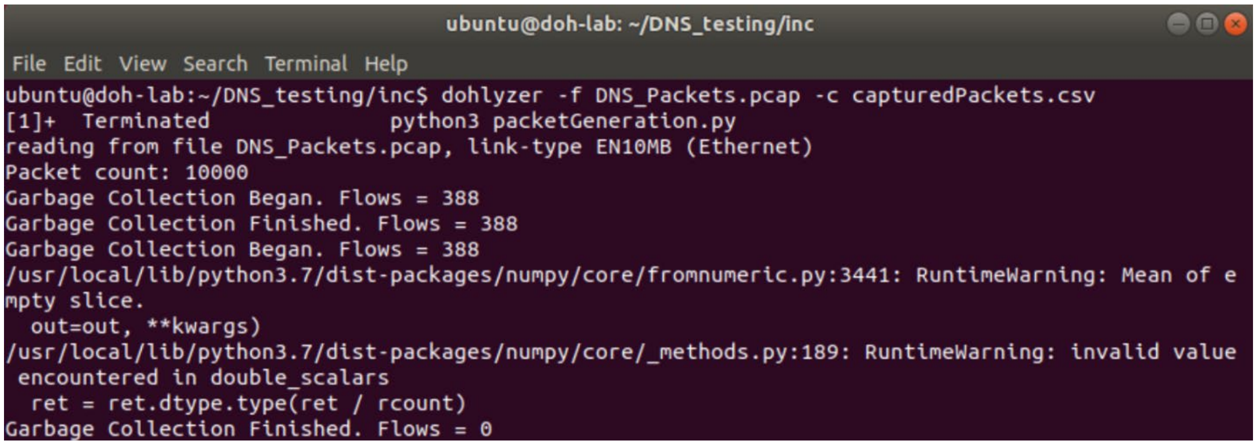
in the directory. This file stores the captured packet information.

g. Pass the DNS_Packets.pcap file through doHlyzer[7]

Use the following command:

dohlyzer -f DNS_Packets.pcap -c capturedPackets.csv

Here, -f specifies the input file, and -c specifies the output .csv file.



ubuntu@doh-lab: ~/DNS_testing/inc
File Edit View Search Terminal Help

```
ubuntu@doh-lab:~/DNS_testing/inc$ dohlyzer -f DNS_Packets.pcap -c capturedPackets.csv
[1]+  Terminated              python3 packetGeneration.py
reading from file DNS_Packets.pcap, link-type EN10MB (Ethernet)
Packet count: 10000
Garbage Collection Began. Flows = 388
Garbage Collection Finished. Flows = 388
Garbage Collection Began. Flows = 388
/usr/local/lib/python3.7/dist-packages/numpy/core/fromnumeric.py:3441: RuntimeWarning: Mean of e
mpty slice.
  out=out, **kwargs)
/usr/local/lib/python3.7/dist-packages/numpy/core/_methods.py:189: RuntimeWarning: invalid value
 encountered in double_scalars
  ret = ret.dtype.type(ret / rcount)
Garbage Collection Finished. Flows = 0
```

g. Run ls again to see a new .csv file.

Once doHlyzer completes its analysis, you'll find a file named

capturedPackets.csv

in the directory. This file presents the packet information in a more readable .csv format.

h. Execute Script:

Execute the

[7] DoHlyzer, a packet analysis tool, is crafted to extract information from an unreadable format (.pcap) and transform it into a more accessible format (.csv), revealing comprehensive details about the packets. While DoHlyzer can capture network traffic, this functionality is unnecessary for this lab.

python3 add-labels.py[8]

script to append labels to each row in the capturedPackets.csv file:

Place in

capturedPackets.csv

as the input and

labeledPackets.csv

as the output

[8] The add-labels script enhances the generated .csv file by appending a new column, indicating whether the website lookup was performed using DNS over HTTPS (DoH) or non-DoH. This is achieved by scrutinizing the IP addresses in the source and destination IP address columns and comparing them with a predefined list. If there is a match, the traffic is identified as DoH. The script accomplishes this task by leveraging the packetGeneration.py script, which generates DoH traffic from two different providers, Google and Cloudflare, each utilizing a couple different IP addresses.

4. Post Lab Exercises

For this next set of questions, you will be analyzing the labeledPackets.csv file . Make sure to keep your lab's VirtualBox VM window open as you'll perform further tasks there.

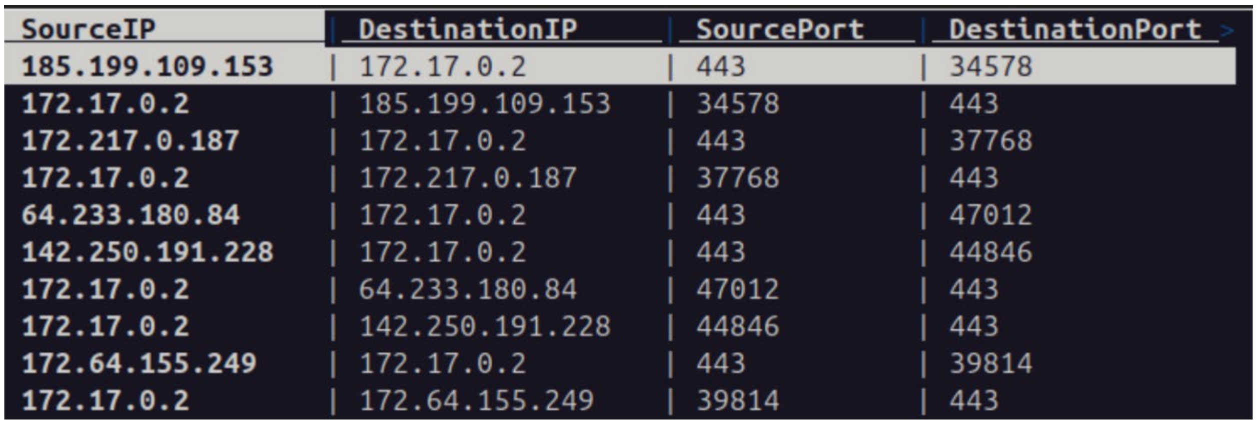
To view the contents of the labeledPackets.csv file, start by installing the VisiData text editor, which displays CSV files in a structured format. Use the following command to install it:

sudo apt install visidata

Once installed, open the labeledPackets.csv file with:

vd labeledPackets.csv

Inside, you'll find a comprehensive collection of packet information captured through tcpdump. This data includes details like source and destination IP addresses, ports, packet length, travel duration, and other packet-specific information. Take a moment to explore this dataset. Towards the end, you'll notice a column labeled 'label' displaying values as either 'non-doh' or 'doh.' This column identifies whether the sent or received packets utilized the DoH (DNS over HTTPS) protocol or the standard non-DoH protocol.

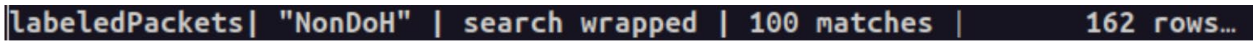


```
SourceIP          | DestinationIP    | SourcePort    | DestinationPort >
185.199.109.153   | 172.17.0.2       | 443           | 34578
172.17.0.2        | 185.199.109.153  | 34578         | 443
172.217.0.187     | 172.17.0.2       | 443           | 37768
172.17.0.2        | 172.217.0.187    | 37768         | 443
64.233.180.84     | 172.17.0.2       | 443           | 47012
142.250.191.228   | 172.17.0.2       | 443           | 44846
172.17.0.2        | 64.233.180.84    | 47012         | 443
172.17.0.2        | 142.250.191.228  | 44846         | 443
172.64.155.249    | 172.17.0.2       | 443           | 39814
172.17.0.2        | 172.64.155.249   | 39814         | 443
```

Tasks:

1. Estimate the approximate percentage of packets that utilized DNS over HTTP (DoH) and the percentage of packets that used regular HTTPS (non-DoH).

   Hint: (Use the row selection | command in the **Label** column alongside NonDoH to count rows with the "NonDoH" label. The result will appear at the bottom of the terminal like this below

   ```
   labeledPackets| "NonDoH" | search wrapped | 100 matches |      162 rows…
   ```

)

2. Identify the IP addresses linked to DoH traffic within the file.

   Hint: (Look at rows that have DoH as their label and check there IP addresses)

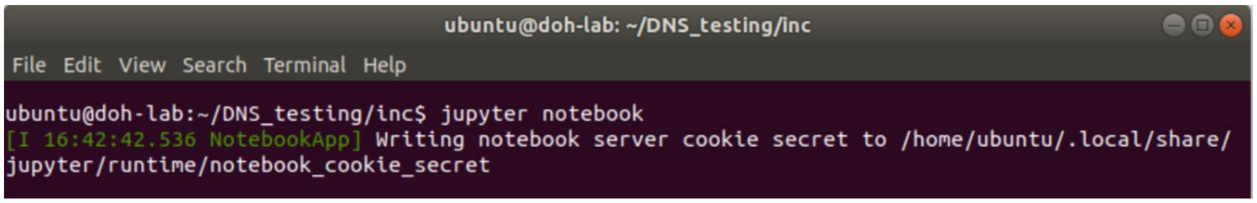
3. Determine the port consistently employed, either as a source or destination. Hint: It corresponds to the well-known port number designated for TCP traffic.

Machine Learning

In this section, you will analyze the 'labeledPackets.csv' file using a decision tree algorithm. Ensure that you are in the 'dohlyzer/inc' folder within the same terminal you performed the lab in. To verify your current location, use the command pwd.

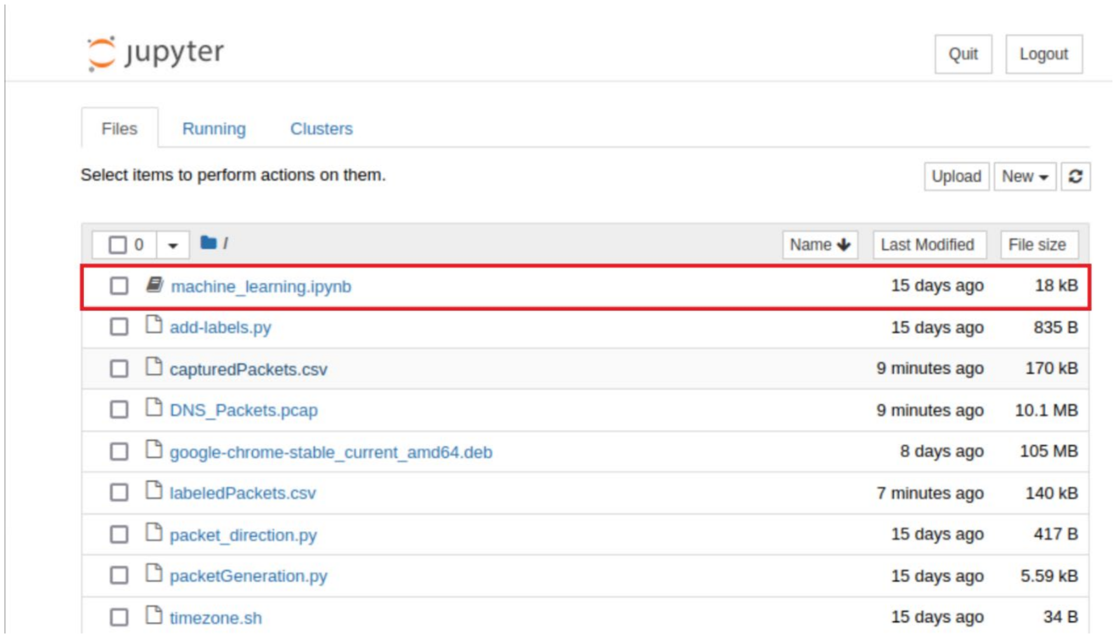
Begin by opening the Jupyter notebook. Type the following command:

jupyter notebook


ubuntu@doh-lab: ~/DNS_testing/inc
File Edit View Search Terminal Help

```
ubuntu@doh-lab:~/DNS_testing/inc$ jupyter notebook
[I 16:42:42.536 NotebookApp] Writing notebook server cookie secret to /home/ubuntu/.local/share/
jupyter/runtime/notebook_cookie_secret
```

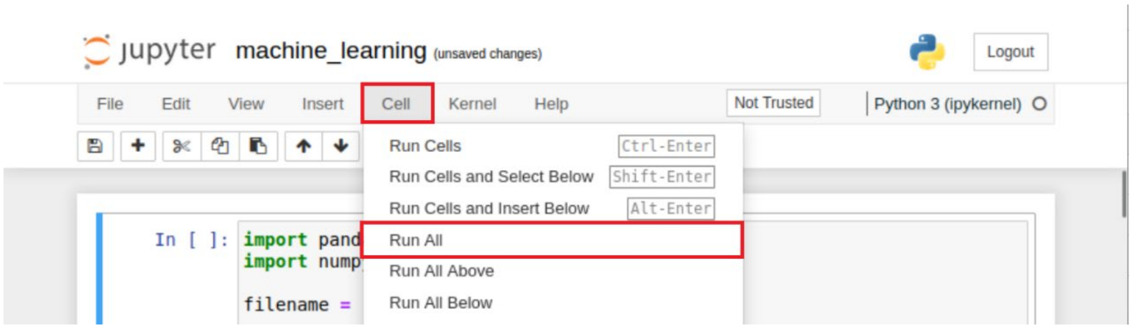
This command will launch a new tab displaying the directory's files. Locate and open the 'machine_learning.ipynb' file. The Jupyter notebook file allows you to read the data from 'labeledPackets.csv' and implement a decision tree algorithm.



Select items to perform actions on them.

Name	Last Modified	File size
machine_learning.ipynb	15 days ago	18 kB
add-labels.py	15 days ago	835 B
capturedPackets.csv	9 minutes ago	170 kB
DNS_Packets.pcap	9 minutes ago	10.1 MB
google-chrome-stable_current_amd64.deb	8 days ago	105 MB
labeledPackets.csv	7 minutes ago	140 kB
packet_direction.py	15 days ago	417 B
packetGeneration.py	15 days ago	5.59 kB
timezone.sh	15 days ago	34 B

Navigate to the 'Cell' menu and select 'Run all' from the dropdown options. This will execute the data processing through the decision tree algorithm.



```
In [ ]: import pand
        import nump

        filename =
```

1. **Note down any intriguing observations or anomalies in the dataset.**

2. **Make a record of the output values after running the file:**

a. Decision tree (unlimited depth)

Accuracy:

Precision:

Recall:

F1 Score:

b. Decision tree (depth 10)

Accuracy:

Precision:

Recall:

F1 Score:

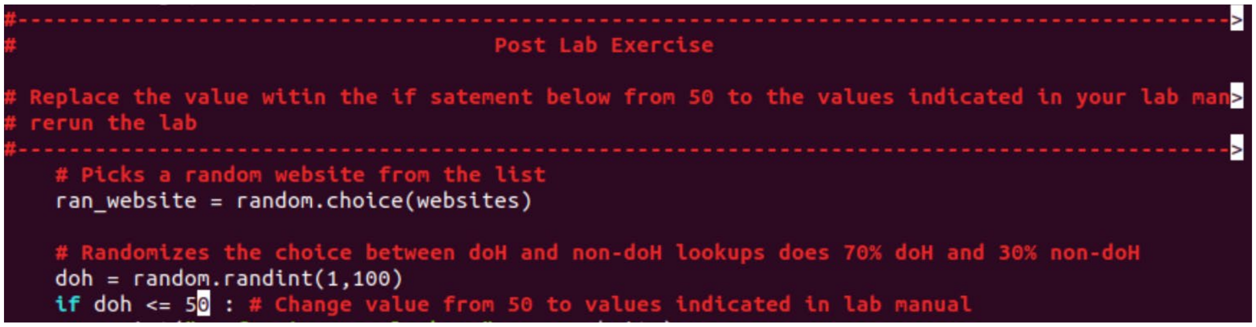
When you are done you can exit the jupyter notebook by using ctrl + c.

Networking

For this next part make sure you are in the 'doh-lab' Labtainer and not the host machine. Ensure you are in the 'dohlyzer/inc' directory. Open the packet generation script 'packetGeneration.py' using the nano text editor:

nano packetGeneration.py

Use the arrow keys to navigate down until you locate "Post Lab Exercise."

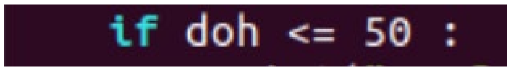


```
#-------------------------------------------------------------------------------------------->
#                                     Post Lab Exercise

# Replace the value witin the if satement below from 50 to the values indicated in your lab man>
# rerun the lab
#-------------------------------------------------------------------------------------------->
    # Picks a random website from the list
    ran_website = random.choice(websites)

    # Randomizes the choice between doH and non-doH lookups does 70% doH and 30% non-doH
    doh = random.randint(1,100)
    if doh <= 50 : # Change value from 50 to values indicated in lab manual
```

The script is set up so that it generates 50% doH traffic and 50% non-doh traffic. Modify the 'if' statement using the values 10, 30, and 80 respectively to increase and decrease the level of doh traffic.



```
if doh <= 50 :
```

After each modification, rerun the lab starting from step d. Analyze the newly generated 'labeledPackets' files each time you input a different value and respond to the questions below. Keep in mind that each rerun will overwrite the 'labeledPackets.csv' file, so make sure to note your results each time since there's no way to revert. For each value (10, 30, 80):

Calculate a rough estimate of DNS over HTTP (DoH) packets vs. DNS packets.

Record the rough estimates for each value:

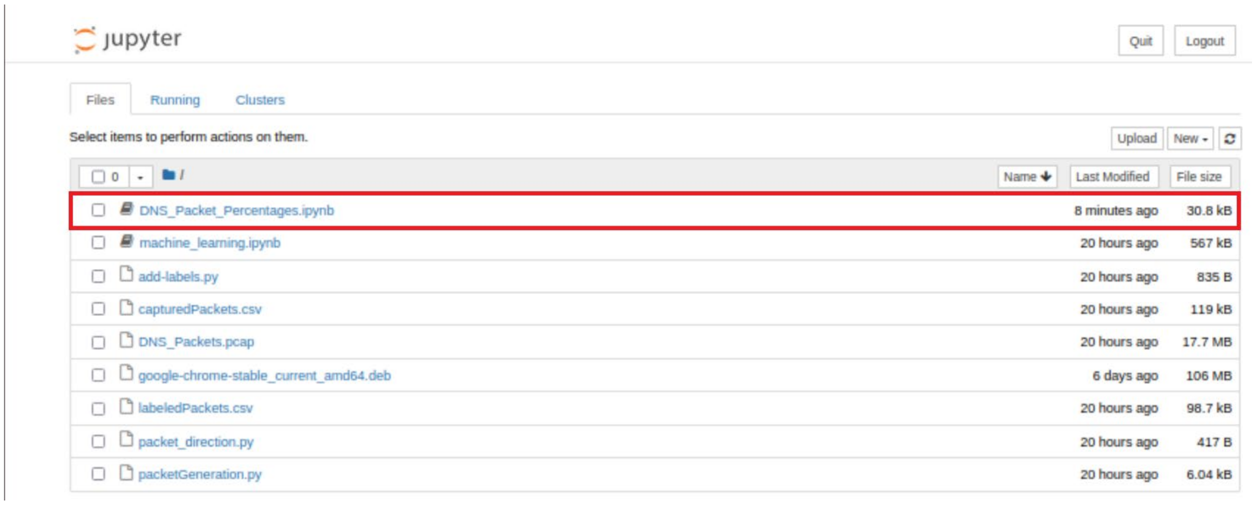
For 10: Provide a Rough Estimate of DoH packets vs. DNS packets. Before proceeding to fill in 30, ensure to answer the first question in the "Analyzing Actual Percentages" section.

For 30: Provide a Rough Estimate of DoH packets vs. DNS packets. Before proceeding to fill in 80, ensure to answer the second question in the "Analyzing Actual Percentages" section.

For 80: Provide a Rough Estimate of DoH packets vs. DNS packets.

Analyzing Actual Percentages

Ensure you are in the 'DNS_testing/inc' directory. Reopen the Jupyter notebook and click on the ‘DNS_Packet_Percentages.ipynb’ file. This will open a Jupyter notebook file that will analyze the percentage of DoH vs. non-DoH traffic in the 'labeledPackets.csv' file. Locate the 'cell' once again and click 'run all.' This will generate a bar graph showing the percentage of DoH and non-DoH packets. Capture a screenshot of the file and insert it below. Repeat for each different value you input into the 'packetGeneration.py' script.



Select items to perform actions on them.

Name	Last Modified	File size
DNS_Packet_Percentages.ipynb	8 minutes ago	30.8 kB
machine_learning.ipynb	20 hours ago	567 kB
add-labels.py	20 hours ago	835 B
capturedPackets.csv	20 hours ago	119 kB
DNS_Packets.pcap	20 hours ago	17.7 MB
google-chrome-stable_current_amd64.deb	6 days ago	106 MB
labeledPackets.csv	20 hours ago	98.7 kB
packet_direction.py	20 hours ago	417 B
packetGeneration.py	20 hours ago	6.04 kB

Screenshot for 10% DoH traffic:

[Insert Screenshot]

Screenshot for 30% DoH traffic:

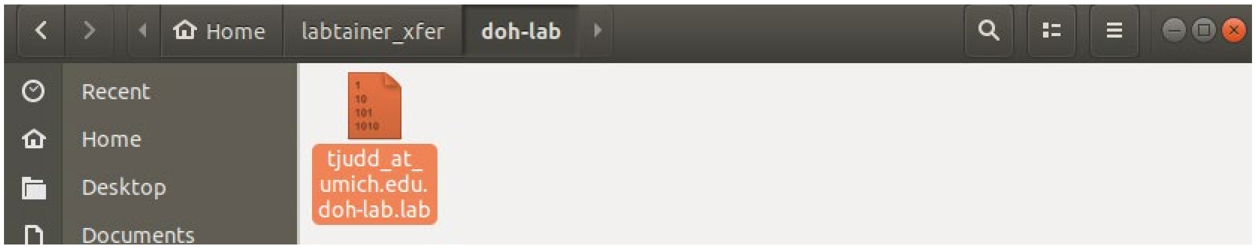
[Insert Screenshot]

Screenshot for 80% DoH traffic:

[Insert Screenshot]

Stopping the Lab and Submitting .xfer file to Canvas

After completing the lab and all post-lab activities, return to the terminal where you initiated the lab. Enter the **'stoplab'** command to halt the lab and close the terminal. Subsequently, locate the file 'labtainer xfer' on your Desktop: Desktop → labtainer xfer → <labname>. Upload this file to Canvas as part of your single zip file, following the specified format: <your_uniquename>_at_umich.edu.doh-lab.lab. Ensure that <your_uniquename> corresponds to your University of Michigan unique name. For instance, mine was named tjudd_at_umich.edu.doh-lab.lab.



A. Preparing the Lab Infrastructure

Preparing the Lab infrastructure - Option #1 : To Run on a Local PC

1. Download and Install VirtualBox: Visit the official VirtualBox website (https://www.virtualbox.org/wiki/Downloads) to download the appropriate version for your computer's operating system.

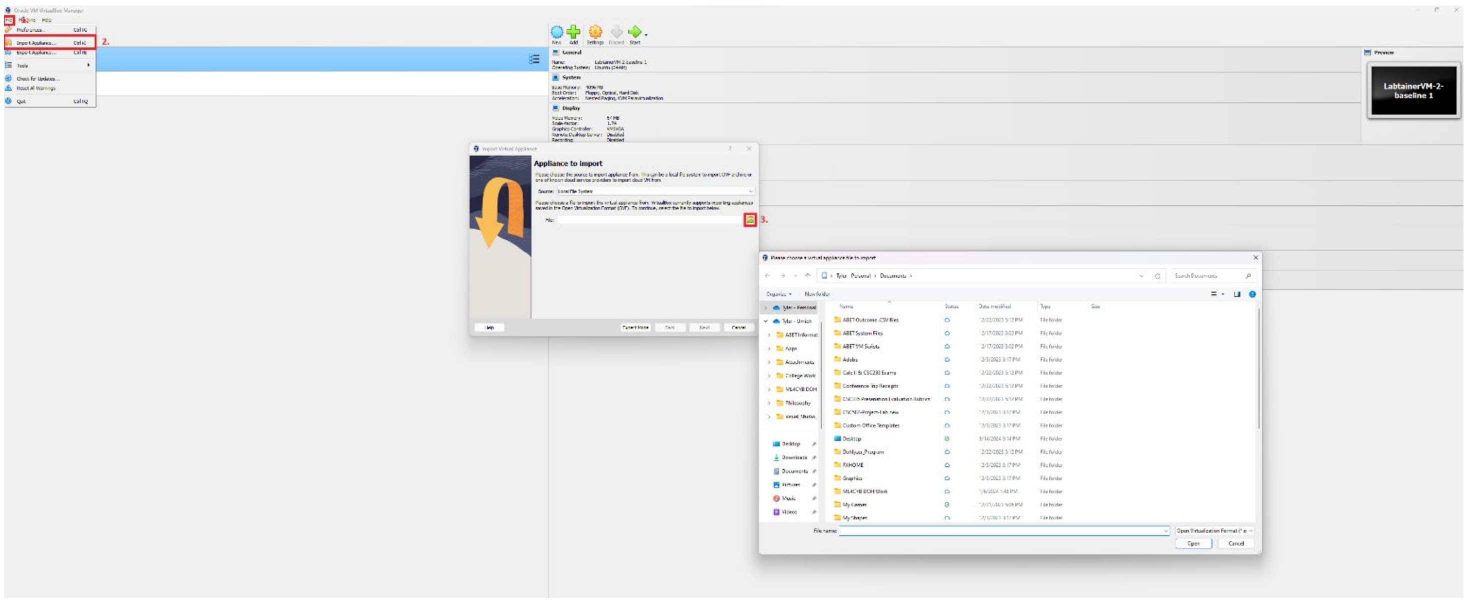
2. Download the VirtualBox VM Appliance: This file is approximately 3.6GB in size, so depending on your internet connection speed, it may take some time to complete the download.

3. Obtain the Labtainer Image: Navigate to https://nps.edu/web/c3o/virtual-machine-images and locate the download link for the VirtualBox VM Appliance. Please note that this file is large and may require several minutes to install.
    a. Import the Labtainer Image into VirtualBox:
    b. Open VirtualBox and go to "File" > "Import Appliance."
    c. Click the small file icon adjacent to the file text box to browse through your directory and select the labtainer image. This process is shown in the figure in the next page.
    d. Alternatively, you can double-click on the Labtainer image file in your directory, which should automatically launch VirtualBox and prompt you to import the image.

4. Start the Virtual Appliance:

5. Once the import process is finished, click on the green left arrow icon within VirtualBox to start the appliance. Please be patient as it may take some time for the virtual machine to start up and become operational.

Note: There are various keyboard shortcuts available for VirtualBox. The Host key is typically set to the right CTRL key. For example, to toggle Full Screen Mode, use Host + F (Right CTRL + F). Refer to Figure 4 for a screenshot of the initial screen after startup.



Please choose the source to import appliance from. This can be a local file system to import OVF archive or one of known cloud service providers to import cloud VM from.

Source: Local File System

Please choose a file to import the virtual appliance from. VirtualBox currently supports importing appliances saved in the Open Virtualization Format (OVF). To continue, select the file to import below.

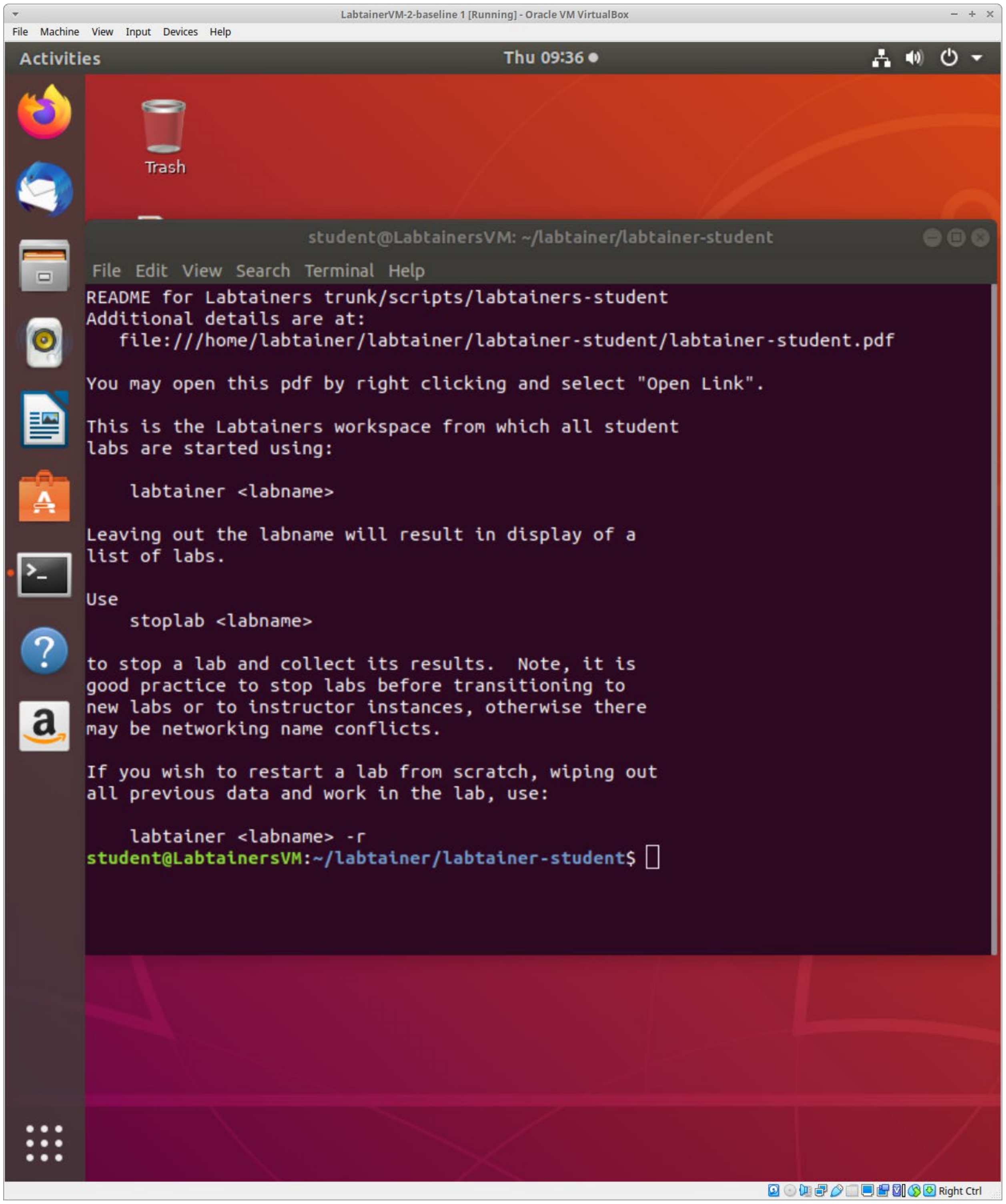
File:

Name: LabtainerVM-2-baseline 1
Operating System: Ubuntu (64-bit)

Base Memory: 4096 MB
Boot Order: Floppy, Optical, Hard Disk
Acceleration: Nested Paging, KVM Paravirtualization

Video Memory: 64 MB
Scale-factor: 1.74
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Importing Labtainer Into VirtualBox


LabtainerVM-2-baseline 1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Thu 09:36
Trash
student@LabtainersVM: ~/labtainer/labtainer-student
File Edit View Search Terminal Help
Right Ctrl

```
README for Labtainers trunk/scripts/labtainers-student
Additional details are at:
   file:///home/labtainer/labtainer/labtainer-student/labtainer-student.pdf

You may open this pdf by right clicking and select "Open Link".

This is the Labtainers workspace from which all student
labs are started using:

    labtainer <labname>

Leaving out the labname will result in display of a
list of labs.

Use
    stoplab <labname>

to stop a lab and collect its results.  Note, it is
good practice to stop labs before transitioning to
new labs or to instructor instances, otherwise there
may be networking name conflicts.

If you wish to restart a lab from scratch, wiping out
all previous data and work in the lab, use:

    labtainer <labname> -r
student@LabtainersVM:~/labtainer/labtainer-student$
```

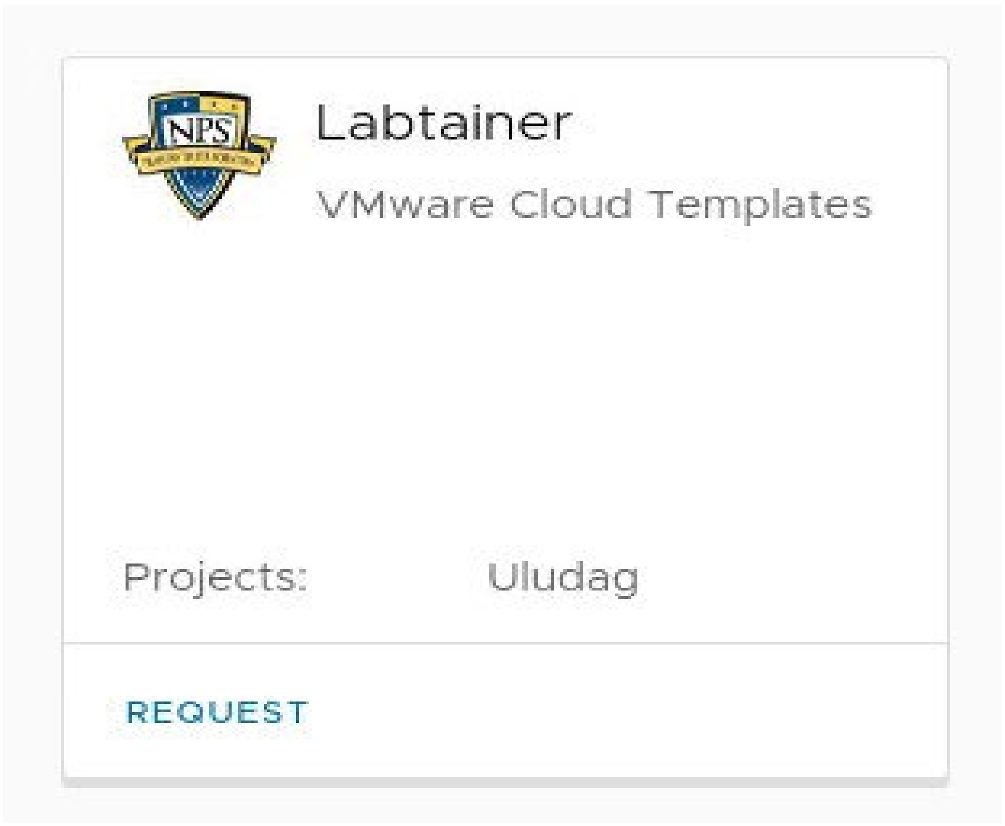
Figure 4: After starting the VirtualBox VM Appliance. Initial screen.

Preparing the Lab infrastructure - Option #2 : To Run on UMFlint vCloud Environment:

Alternative to the above option of running the labtainers on your own PC is to run it over the UM-Flint vCloud Environment to access using your browser.

1. You must be using UM-Flint VPN if not accessing it from campus: vpn.umflint.edu
   Please contact ITS for help in setting up VPN. More information about VPN is at https://teamdynamix.umich.edu/TDClient/99/Portal/KB/ArticleDet?ID=6023.

2. Use the instructions given in the following pages. Request **Labtainer** catalog item from the VMware Cloud Templates, as shown below:, not the Bio Informatics shown in the following page.



Deploying Virtual Machines to the UM Flint vCloud Environment

You will be using the UM Flint vCloud environment to deploy your virtual machine (VM). You will interact with two parts of a service broker to manage your deployments, the catalog and deployment sections.

Service Broker	
Catalog	**Deployments**
This section contains all of the VM templates your computer science instructors have given you access to. If you are enrolled in multiple classes that use this system you will see templates for all of your classes here.	This section contains all of your active and pending deployments. You can view information about and interact with your deployed virtual machines here.

VPN Required

Remote access to the vCloud environment is only available over the UM Flint VPN. On campus systems can directly connect. Configuring a VPN connection on your system is outside the scope of this document, more information can be found here: https://support.umflint.edu/its/article/using-vpn-resources-at-um-flint-2

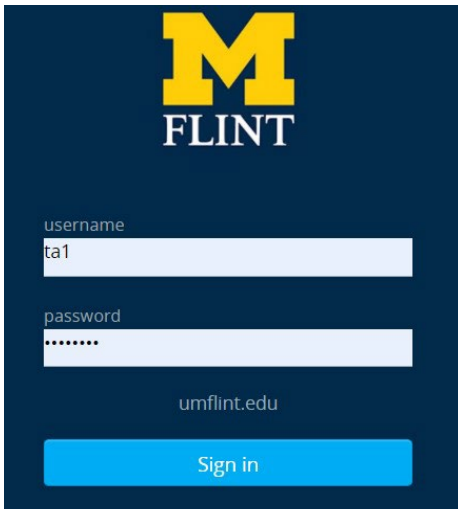
If you are working remotely, establish a VPN connection the university to continue with the rest of this guide.

Login

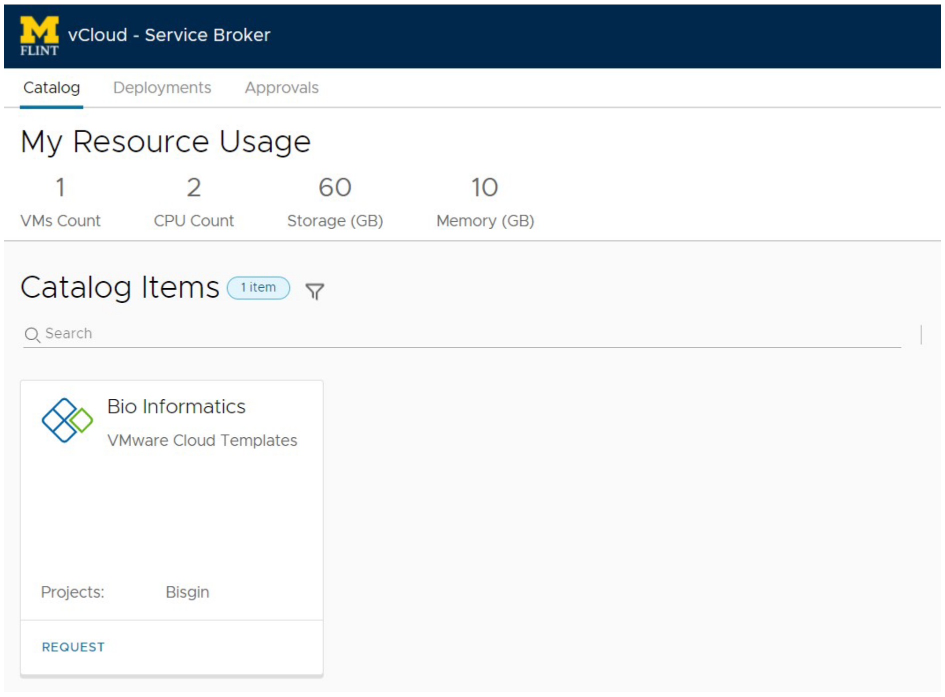
Go to: https://vcloud.umflint.edu/catalog/#/library

If this is the first time you are accessing the portal be sure the "umflint.edu" domain is selected and click the "Next" button.

Sign in to the portal using your unique name (the first part of your email address without the @umich.edu part) and your password. Be sure the domain is listed as "umflint.edu" under the password box.



You will be taken to the "Catalog" section of the service broker. Here you will see a listing of the templates you have access to. The instructor that issued the template is listed next to the template "Projects" field.


vCloud - Service Broker

My Resource Usage

1	2	60	10
VMs Count	CPU Count	Storage (GB)	Memory (GB)

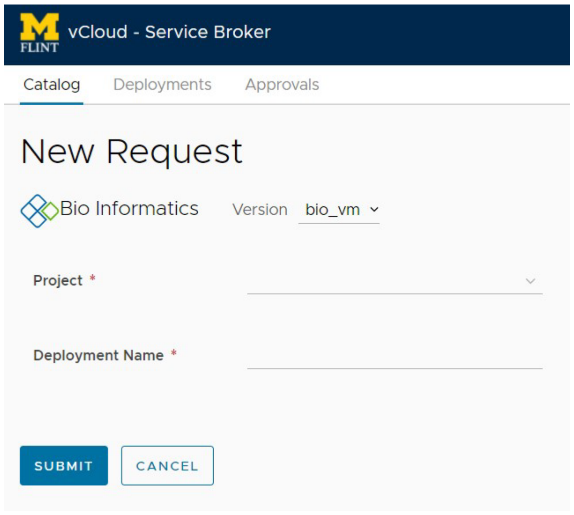
Catalog Items 1 item

Bio Informatics
VMware Cloud Templates

Projects: Bisgin

Deploy

To deploy a desired template click on the “REQUEST” link on the bottom of the template.


vCloud - Service Broker

New Request

Bio Informatics Version bio_vm

Project *

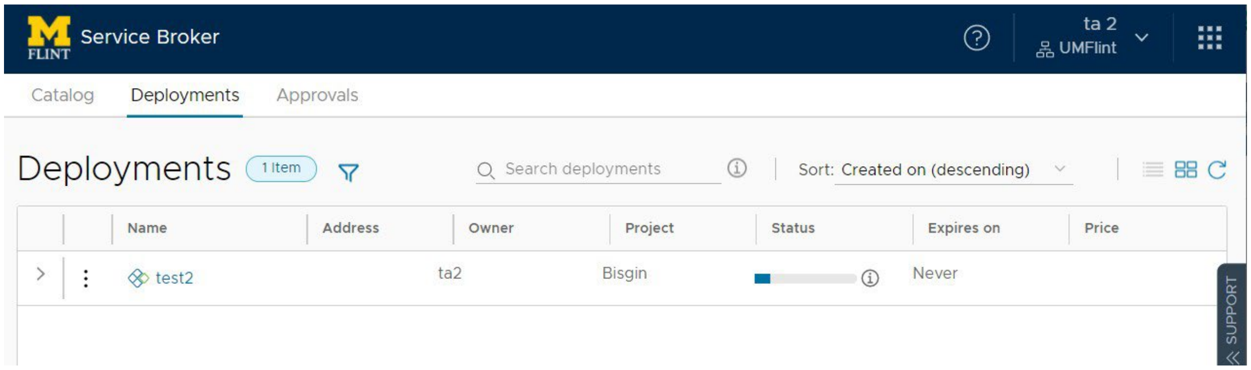
Deployment Name *

The latest template version will be selected for you from the drop down “Version” menu.

Select the instructor name under the “Project” drop down menu.

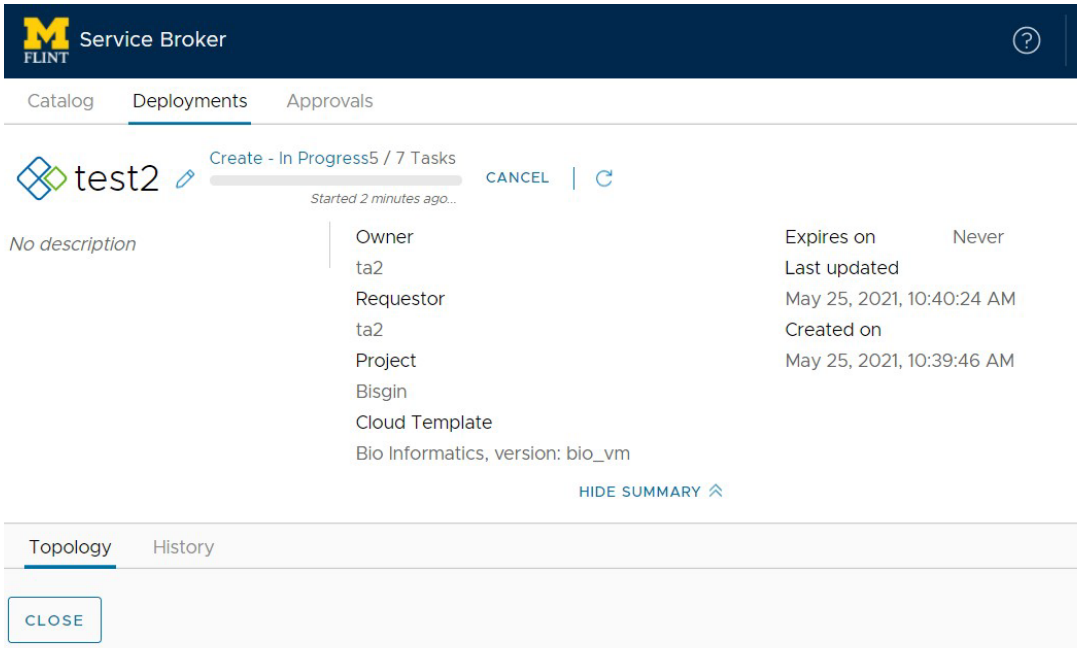
Give your deployment a descriptive name in the “Deployment Name” field. Try to include your course number, unique name (user ID), or other information to help your instructor identify the purpose of your deployment.

Click “Submit”.



Name	Address	Owner	Project	Status	Expires on	Price
test2		ta2	Bisgin		Never	

In the deployment section you will see the new deployment listed. You can click on the deployment name to see the progress of the deployment.



No description

Owner
ta2
Requestor
ta2
Project
Bisgin
Cloud Template
Bio Informatics, version: bio_vm

Expires on Never
Last updated
May 25, 2021, 10:40:24 AM
Created on
May 25, 2021, 10:39:46 AM

The topology section on the bottom shows a map of what is being created and deployed as part of your request.

Be patient. It may take 15 minutes or more for the system to clone the VM and deploy it to your workspace. This time may differ based on the template size, complexity, and other deployment requests in the queue.

Accessing the VM

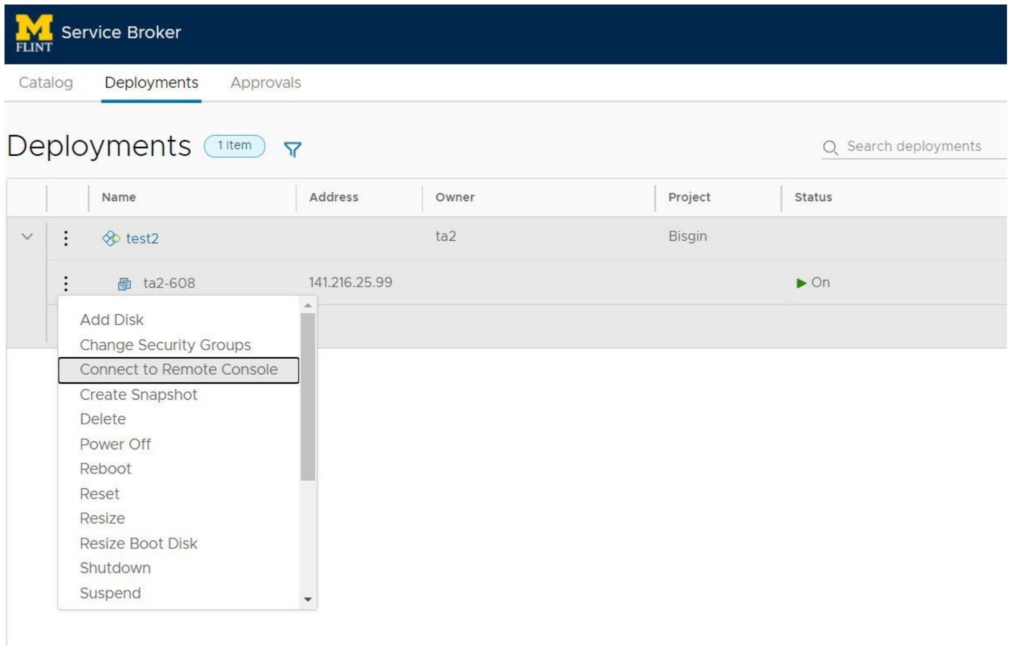
Once a deployment has finished, there are two primary ways you can interact with your VM. Once access is configured, you will probably interact with your VM most often through the use of an integrated or third party connection tool. Depending on the operating system and how your instructor has configured your template, the tool may be SSH for a Linux system, RDP for a windows system, or a TeamViewer session. Your instructor may have more information on the best method for connecting to your particular VM.

The second way you can interact with your VM is using the console from the deployment menu. The console is a representation of what you would see sitting in front of the screen if the VM was a physical machine.

Most VMs will power on once they have deployed. In the "Deployments" section expand a deployment using the arrow to the left of the deployment you want to interact with, you will see the network connection and the VM listed in the deployment.

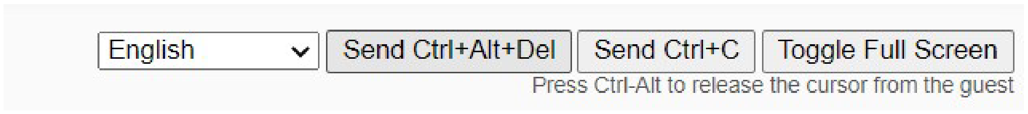
Notice that the VM will have the IP address listed next to the VM name as 141.216.XX.XXX. This is the IP address you can use for RDP or SSH access to the VM (if enabled). If you are using a third party tool you may need to access the VM console first to get the session's configuration information.

Left click the three dots to the left of the VM to pop up a menu, from here you can choose "Connect to Remote Console". This will open a new browser tab where you can interact with the VM console.



Name	Address	Owner	Project	Status
test2		ta2	Bisgin	
ta2-608	141.216.25.99			▶ On

On the console's browser page, buttons should be available in the upper right corner to pass special key combinations to the VM.



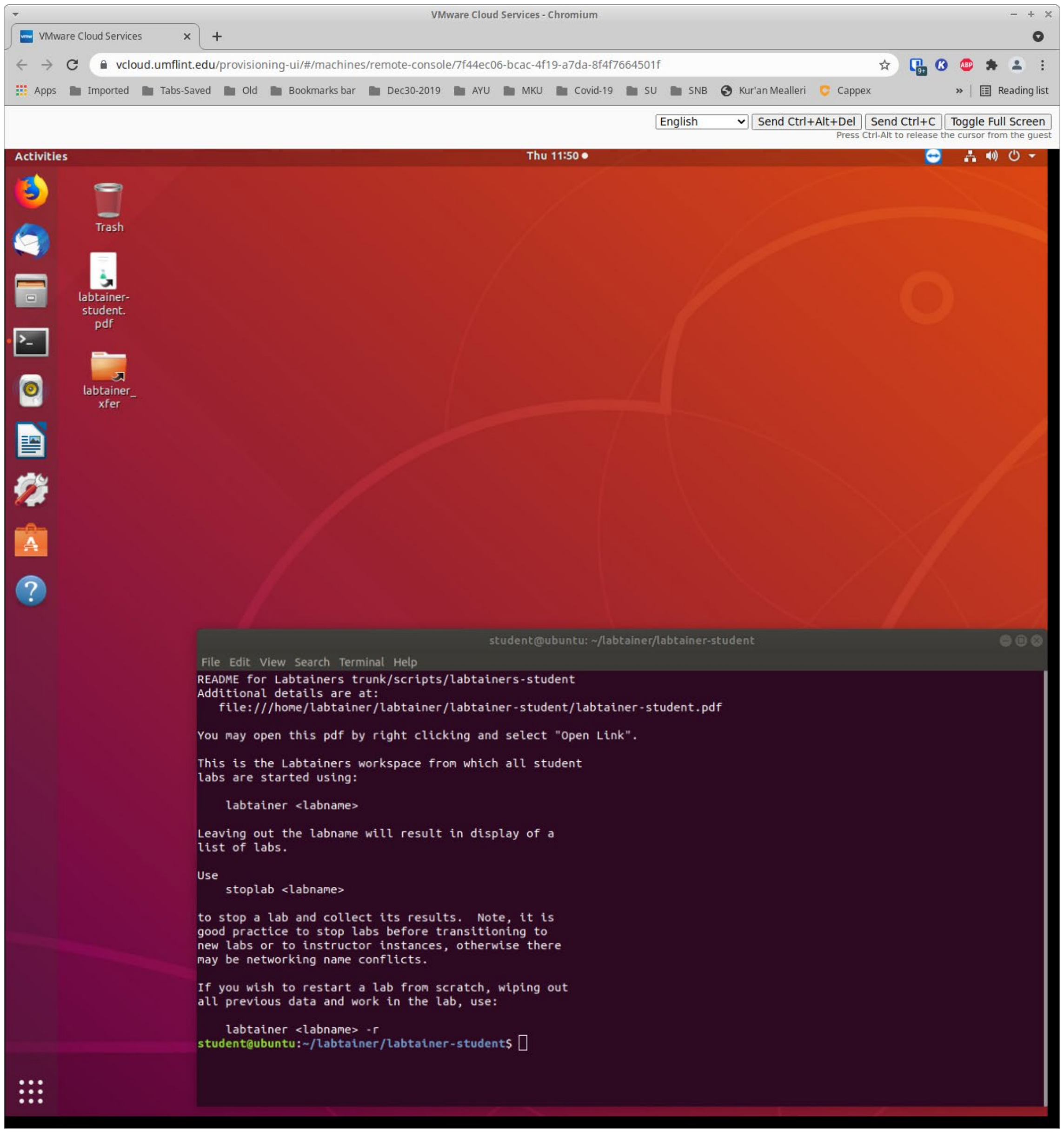
Your instructor will provided you with credentials to log onto your VM. Now would be a good time to log on to the VM and change the default credentials to something more secure.

Tips

If you forget the direct URLs for accessing the catalog or your deployment you can simply use https://vcloud.umflint.edu/ Although, you will need to click through a couple of extra prompts to log into the system and choose the service broker.

A VPN connection is required if you are using SSH or RDP to connect to your VM.

After choosing the Connect to Remote Console, you will get the Ubuntu VM access as shown in Figure 5.


VMware Cloud Services - Chromium
VMware Cloud Services
vcloud.umflint.edu/provisioning-ui/#/machines/remote-console/7f44ec06-bcac-4f19-a7da-8f4f7664501f
Apps
Imported
Tabs-Saved
Old
Bookmarks bar
Dec30-2019
AYU
MKU
Covid-19
SU
SNB
Kur'an Mealleri
Cappex
Reading list
English
Send Ctrl+Alt+Del
Send Ctrl+C
Toggle Full Screen
Press Ctrl-Alt to release the cursor from the guest
Activities
Thu 11:50
Trash
labtainer-student.pdf
labtainer_xfer
student@ubuntu: ~/labtainer/labtainer-student
File Edit View Search Terminal Help

```
README for Labtainers trunk/scripts/labtainers-student
Additional details are at:
   file:///home/labtainer/labtainer/labtainer-student/labtainer-student.pdf

You may open this pdf by right clicking and select "Open Link".

This is the Labtainers workspace from which all student
labs are started using:

    labtainer <labname>

Leaving out the labname will result in display of a
list of labs.

Use
    stoplab <labname>

to stop a lab and collect its results.  Note, it is
good practice to stop labs before transitioning to
new labs or to instructor instances, otherwise there
may be networking name conflicts.

If you wish to restart a lab from scratch, wiping out
all previous data and work in the lab, use:

    labtainer <labname> -r
student@ubuntu:~/labtainer/labtainer-student$ 
```

Figure 5: vCloud access to labtainers VM from within a browser (Chromium) over VPN.

Preparing the Lab infrastructure - Option #3 : To Run on the cloud free (Azure or GCP) :

Labtainers can be run on cloud services and accessed via a browser. Cloud service providers may offer free accounts for students or others looking to learn about their cloud services. Currently, Labtainers works with the Azure and Google cloud platforms as described below.

A.1 Azure Cloud

These instructions assume the user has an Azure account, e.g., https://azure.microsoft.com/en-us/free/students/

This approach requires that the Azure CLI be installed on the Mac, Windows or Linux: https://docs.microsoft.com/en-us/cli/azure/install-azure-cli

In the following command examples, use the ”ps1” file extension instead of ”sh” when using PowerShell.

Open a terminal on Mac/Linux, or a PowerShell window on Windows.

Install the local scripts by getting this script (make it executable on Mac or Linux): https://raw.githubusercontent.com/mfthomps/Labtainers/master/azure/install_labtainers.sh Or on Windows: https://raw.githubusercontent.com/mfthomps/Labtainers/master/azure/install_labtainers.ps1 On Mac or Linux:

- curl -L https://raw.githubusercontent.com/mfthomps/Labtainers/master/azure/install_labtainers.sh --output install labtainers.sh
- chmod a+x install labtainers.sh

On Windows:

- wget https://raw.githubusercontent.com/mfthomps/Labtainers/master/azure/install_labtainers.sh -OutFile install _labtainers.ps1

Then run it (Mac/Linux).

```
./install_labtainers.sh
```

Windows:

```
./install_labtainers.ps1
```

That will create a $HOME/labtainers azure directory.

Change to the $HOME/labtainers azure directory

```
cd $HOME/labtainers_azure
```

Log into your Azure account:

```
az login
```

NOTE: If your account has access to more than one Azure Subscription, you need to change these parameters to specify the student subscription before running the install labtainers script:

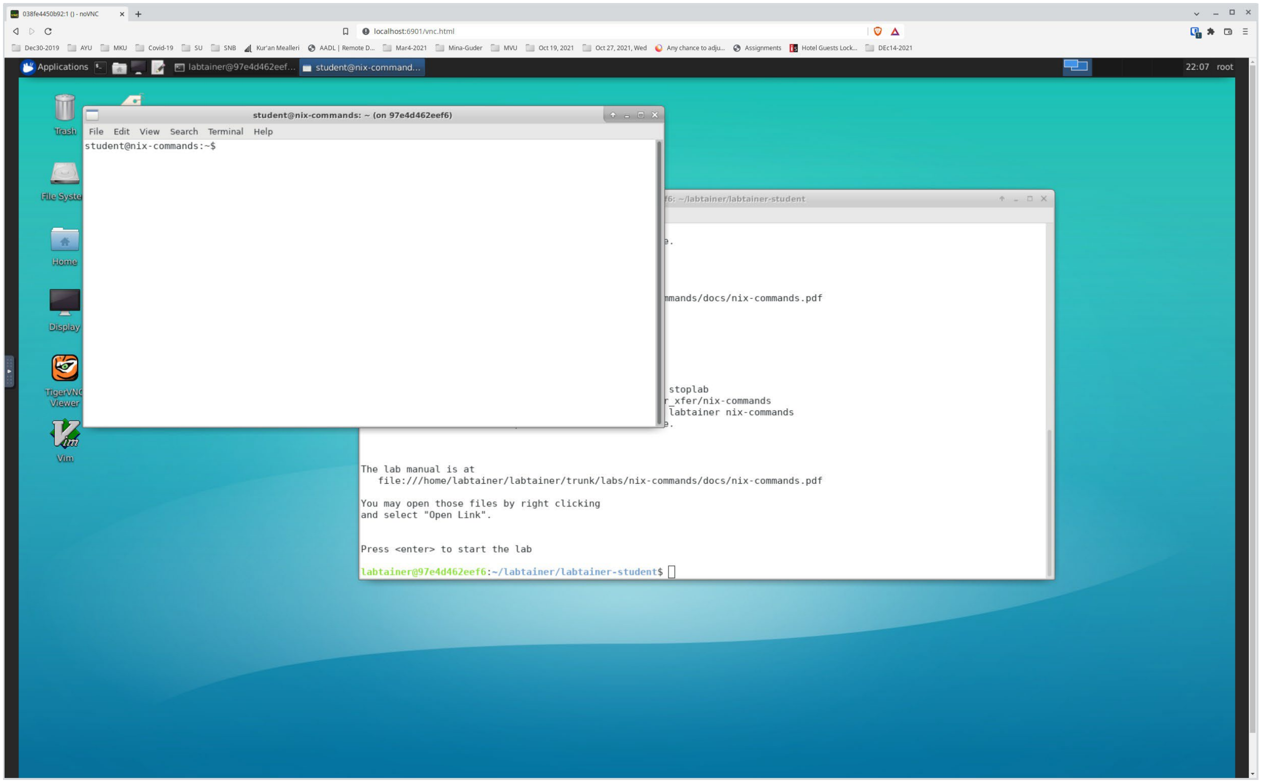
1. Change the /.azure/clouds.config to show your student subscription number
2. Change the entries in /.azureProfile.json so that only your student subscription shows “isDefault”= true, the rest being set to “false”.

Once logged into Azure, run the create vm.sh (or create vm.ps1 for windows) script, passing in a user ID. The ID can be any name, e.g.,

```
./create_vm.sh myname
```

The create vm script may take a while to run. The process is complete when you see *Labtainers is up*. Point a local browser to localhost:6901 and perform the labs. When prompted for a password in the browser, just click submit or OK, i.e., leave the password blank. The password for the *labtainer user* in the VM is “labtainer”.

Figure 6 shows a screenshot of my access to Azure after using their free account. Please make sure to deallocate and delete not to use up your free access allocation. See details below.



```
student@nix-commands:~$
```

```
The lab manual is at
   file:///home/labtainer/labtainer/trunk/labs/nix-commands/docs/nix-commands.pdf

You may open those files by right clicking
and select "Open Link".


Press <enter> to start the lab

labtainer@97e4d462eef6:~/labtainer/labtainer-student$
```

Figure 6: Cloud access for labs using the Azure free allocation from within your browser.

Select and perform the lab as needed. Then refer to the items below.

When done with labs, run the get results.sh (or get results.ps1) script:

```
./get_results.sh <user ID>
```

This will store your Labtainer results in $HOME/labtainer xfer. Provide those results to your instructor.

If you become unable to reach the Labtainers via your browser, e.g., after shutting down your computer, use the restart.sh script:

```
./restart.sh <user ID>
```

The create vm.sh script will create an SSH key pair named id labtainers within your $HOME/.ssh directory. The private key in id _labtainers is not passphrase protected, so you must protect it. You may move the keys to a different computer and access your Labtainers from that computer's browser. You must first run the install labtainers.sh script on that computer, and then run the restart.sh script.

When done with a lab, use

```
./deallocate_vm <user ID>
```

to stop incurring most charges. Note however that any work you've performed on the Labtainers might be lost (unless you've retrieved your results with get results.sh), depending on how long the VM is dormant.

To restore a VM after you deallocated it, use:

```
./restore_vm.sh <user ID>
```

When completely done with the VM, use the delete vm.sh script to stop incurring all charges:

```
./delete_vm.sh <user ID>
```

Shutting down the VM without deallocating or deleting it will not stop charges.

A.2 Google Cloud Platform

These instructions assume you have a google cloud account. https://cloud.google.com/ This requires that the Google Cloud SDK be installed on the Mac, Windows or Linux: https: //cloud.google.com/sdk/docs/quickstart

On Linux/Mac, add the google-cloud-sdk/bin directory to your PATH environment variable. For example, if you put the SDK in your home directory, then add this to your $HOME/.bash profile

PATH=$PATH:$HOME/google-cloud-sdk/bin and then run source $HOME/.bash_profile

On Windows, just reopen a new PowerShell window after installing the SDK.
In the following command examples, use the "ps1" file extension instead of "sh" when using PowerShell.

Open a terminal on Mac/Linux, or a PowerShell window on Windows.

Install the local scripts by getting this script (make it executable on Mac or Linux): https://raw.githubusercontent.com/mfthomps/Labtainers/master/google/install_labtainers.sh Or on Windows: https://raw.githubusercontent.com/mfthomps/Labtainers/master/google/install_labtainers.ps1 On Mac or Linux:

- curl -L https://raw.githubusercontent.com/mfthomps/Labtainers/master/google/install_labtainers.sh --output install labtainers.sh
- chmod a+x install labtainers.sh

On Windows:

- wget https://raw.githubusercontent.com/mfthomps/Labtainers/master/google/install_labtainers.sh -OutFile install labtainers.ps1

Then run it (Mac/Linux).

```
./install_labtainers.sh
```

Windows:

```
./install_labtainers.ps1
```

That will create a $HOME/labtainers google directory.

Change to the $HOME/labtainers google directory

```
cd $HOME/labtainers_google
```

Log into your Google cloud account from the command line:

```
gcloud auth login
```

Define your default region and zone by editing and running the set defaults.sh script. And then initialize using:

```
gcloud init
```

Once logged into the Google Cloud with default region/zone defined, run the create vm.sh
(or create vm.ps1 for windows) script, passing in a user ID. The ID can be any name, e.g.,

```
./create_vm.sh myname
```

On Linux/Mac, you will be prompted for an ssh passphase, leave it blank. On Windows, ignore the warnings about ssh keys.

The create _vm script may take a while to run. The process is complete when you see "Labtainers is up. Point a local browser to http://localhost:6901" and perform the labs. When prompted for a password in the browser, just click submit or OK, i.e., leave the password blank. The password for the labtainer user in the VM is labtainer.

When done with labs, run the get results.sh (or get results.ps1) script:

```
./get_results.sh <user ID>
```

This will store your Labtainer results in /labtainer xfer. Provide those results to your instructor.

If you become unable to reach the Labtainers via your browser, e.g., after shutting down your computer, simple use the restart.sh script:

```
./restart.sh <user ID>
```

The create vm.sh script will create an SSH key pair named id labtainers within your /.ssh directory. The private key in id _labtainers is not passphrase protected, so you must protect it. You may move the keys to a different computer and access your Labtainers from that computer's browser. You must first run the install labtainers.sh script on that computer, and then run the restart.sh script.

When done with a lab, use

```
./stop_vm.sh <user ID>
```

to stop incurring processing charges. Note you may still incur storage charges until the VM is delete.

To restore a VM after you stopped it, use:

```
./start_vm.sh <user ID>
```

When completely done with the VM, use the delete vm.sh script to stop incurring all charges:

```
./delete_vm.sh <user ID>
```

Shutting down the VM without deleting it will not stop all charges, but will stop processing charges. See the Google Cloud dashboard and pricing for more information.

```
sudo chmod 755 /var/lib/dpkg/lock
```