

CSC 535

Advanced Computer Networking

University of Michigan-Flint



COMPUTER SCIENCE

Fall 2022

October 31, 2022

Homework 4

(100 points)

due by Nov 7, Mon, 11:30am Eastern Time

Remarks:

- Zip all the required files for your homework into one file with the following format: **LastName-Firstname-535-hw4.zip**. For example, for me it would be **Uludag-Suleyman-535-hw4.zip**. 10% penalty for not following the zip file name convention. 10% penalty for uploading multiple files.
 - No emailed homeworks will be accepted.
 - Only submission is via the Canvas system.
 - No late submissions will be accepted.
 - No submission means automatic 0.
 - **Individual submission, not a group work! You cannot share your answers with anyone in the class. Both sharing and receiving will get zero and be reported to the university administration to seek the full invocation of the academic integrity violation rules.**
 - Refer to Appendix [A](#) for three different ways of setting up your lab infrastructure through one of the three options; local, UM-Flint vCloud, or Cloud (Azure and GCP).
-

Questions for the deliverable:

1. `tracert` (in Windows) and `tracpath` (in Ubuntu Linux) program is a network tool that reveals the Internet path from the source to the destination together with the delay values. An example run is given below in Ubuntu Linux:

```
suleyman@suleyman-ThinkPad-W520:~$ tracpath cnn.com
 1:  msb11-53.flint.umich.edu                0.170ms pmtu 1500
 1:  msb11-1.flint.umich.edu                 56.364ms
 1:  msb11-1.flint.umich.edu                 71.110ms
 2:  xe-1-0-1-r-msb-fw.its.flint.umich.edu   55.108ms asymm  3
 3:  cg-bb.umflint.edu                       55.117ms
 4:  vlanx208.um-fln-msb.mich.net            63.746ms
 5:  irbx87.glc-fln.mich.net                 95.644ms asymm 11
 6:  irbx59.glc-lan3.mich.net                63.715ms asymm 10
 7:  xe-0-0-0x20.msu6.mich.net               63.695ms asymm 10
 8:  xe-4-1-0x14.eq-chi2.mich.net            68.700ms
 9:  12.250.16.17                            71.358ms asymm 10
10:  cr2.cgcil.ip.att.net                    60.821ms asymm 12
11:  gar13.cgcil.ip.att.net                  57.968ms
12:  no reply
13:  4.69.158.146                           93.123ms asymm 20
14:  ae-5-5.ebr2.chicago1.level3.net        110.660ms asymm 20
15:  ae-3-3.ebr2.atlanta2.level3.net          102.042ms asymm 19
16:  ae-2-52.edge4.atlanta2.level3.net       103.308ms asymm 18
17:  no reply
18:  no reply
19:  no reply
20:  no reply
21:  no reply
22:  no reply
23:  no reply
24:  no reply
25:  no reply
26:  no reply
27:  no reply
28:  no reply
29:  no reply
30:  no reply
31:  no reply
    Too many hops: pmtu 1500
    Resume: pmtu 1500
suleyman@suleyman-ThinkPad-W520:~$
```

Explain how the `tracpath`/`tracert` works in details after researching about it and cite all your sources properly and fully. No copy-paste please. Paraphrasing is OK as well as some limited quoting.

2. Pick 5 web sites: One from North America, one from Europe, one Asia, one Africa and one from South America. Use `tracert` to these sites 4 times, at least on 3 different days. Turn in the outputs as part of your answers. Did the `tracert` output for the same sites generate identical routes for all the sites? Are the outputs supposed to be identical? Discuss why or why not. Also, plot the delay values for all these 5 sites for 4 runs on 5 separate graphs. If the sites are not reporting and timing out, please use the latest reported time as the delay value. X-axis will be the run # and Y will be the delay value. For example, my data from the previous questions will have a single data point of $103.308ms$ for the y-axis.
3. Briefly summarize the categories of the ISPs as discussed in class.
4. Briefly explain the standardization process for the Internet related protocols.
5. Define the network parameters discussed in class and briefly explain them with their units of measurements.
6. What are the four initial Internetting principles from Leiner, et al paper? Briefly discuss them. (Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Larry G. Roberts, and Stephen Wolff. 2009. A brief history of the internet. SIGCOMM Comput. Commun. Rev. 39, 5 (October 2009), 22–31. DOI: <https://doi.org/10.1145/1629607.1629613>). The paper is posted at canvas.flint.umich.edu.

Lab #1: Unix users and groups

(a) Please complete the lab below.

1 Overview

This exercise introduces management of users and groups on a Unix system. The lab includes the following objectives:

- Add users to a shared system.
- Define a group on the system, and assign users to that group.
- Observe how user and group IDs can affect access to files.
- Observe a limitation of discretionary access controls.
- Grant a user *sudo* or *superuser* privileges.

This lab does not provide in-depth coverage of users, groups and Unix permissions. It introduces the concepts and provides hands-on experience. Use the Unix help facilities, e.g., `useradd -h` to view detailed options for different commands covered in this lab.

1.1 Background

This is an introductory lab that only requires a bit of familiarity with the Unix command line.

2 Lab Environment

This lab runs in the Labtainer framework, available at <http://nps.edu/web/c3o/labtainers>. That site includes links to a pre-built virtual machine that has Labtainers installed, however Labtainers can be run on any Linux host that supports Docker containers.

From your labtainer-student directory start the lab using:

```
labtainer users
```

A link to this lab manual will be displayed.

3 Environment

While interfaces for adding and managing users may vary between installations, the concept of Unix users and groups are the same across all Unix and Linux systems. In this lab you will manage a CentOS7 system using the *useradd* command.

When the lab starts, two virtual terminals appear, each with a standard Unix login prompt. Both terminals connect to the same computer, called *shared*. Initially, there is one user named *admin* defined, with a password of *password123*.

Login to one of the terminals as *admin* and use `sudo su` to get a root shell. Use that root shell to manage users for the lab. Use the other terminal to test those users per the lab directions below.

4 Tasks

4.1 Add user Bob

Use this command to add a user with id `bob`:

```
useradd -m bob
```

The `-m` option causes a home directory to be created for the user at `/home/bob`. Go to the other login terminal and try to login as the new user `bob`. Oh, that's right, the user does not yet have a password. Passwords are assigned and changed using the `passwd` program. Use:

```
man passwd
```

to learn about options for the `passwd` program. For this lab, we'd like to assign bob a password so that bob can login, but we want the password to be expired so that bob is forced to change it. Use the `passwd` program to assign bob a password, and then run the `passwd` program to cause bob's password to be expired. Then login as bob and confirm that bob is required to change the password. Do that and make note of the password for later. (For this lab it is **OK** to write down passwords!)

Deliverable#1: (Screenshot) Provide a screenshot of login to the shared machine using the username `bob` and changing your password.

Then use `exit` to log bob out of the terminal.

4.2 Add user Mary

Add a user with id `mary` just like you added bob. Before you try to login as Mary, take a look at a file that Mary will need to access. Use this command:

```
ls -l /shared_stuff/tarts.txt
```

Note who owns the file, and the permitted access modes for the owner, the group and other.

```
-rw-rw---- 1 frank bakers 8 Apr 29 23:48 /shared_stuff/tarts.txt
```

A brief tutorial on Unix file permissions can be viewed at: <https://mason.gmu.edu/~montecin/UNIXpermiss.htm>. For this file, the owner has read and write permission, as do members of the group. Users other than the owner or members of the group have no access to the file. The owner is `frank` and the group is `bakers`. The permissions indicate that only members of the `bakers` group should be able to read or write the file. We want Mary to be able to access this file because she is a baker. So, add Mary to the `bakers` group. This is done with the `usermod` command:

```
usermod -a -G bakers mary
```

Now login to the terminal as `mary`. Then use the `id` command to confirm that `mary` is in the `bakers` group. Then confirm that Mary can view the file:

```
cat /shared_stuff/tarts.txt
```

Deliverable#2: (Screenshot) Provide a screenshot of your current terminal with the output of the cat program visible.

Now run one of the bakers' application programs:

```
eggcheck tarts.txt
```

When the eggcheck program runs, it runs as mary with her permissions. This lets the program read the file.

Use the `id` command again. Note how the user is mary and the group is mary. While mary is a member of both the mary group and the bakers group, her current id reflects the mary group. Use this command to create a new file as mary:

```
touch newfile.txt
```

Use `ls -l` to view ownership of the file. Then change Mary's current group by using:

```
newgrp bakers
```

and use `ls -l` again. Many things can affect the permissions of a file, including the current group of the user creating the file.

Deliverable#3: (Screenshot) Provide a screenshot of your current terminal.

4.3 Re-login as bob

Use `exit` to exit the mary session and login as bob again. Try to view the tarts.txt file:

```
cat /shared_stuff/tarts.txt
```

Since Bob is not a member of the bakers group, Bob is not able to access the file. This type of control over access to files is called *discretionary access control* because it relies on the discretion of the members of the bakers group. For example, at their discretion, a member of the bakers group could create a copy of the tarts.txt file that is visible to users who are not in the bakers group. However, it is not only the discretion of the users, but also the discretion of the *programs* executed by those users. Recall Mary ran the `eggcheck` program. That program happens to create a temporary copy of the file that it reads, and places that copy in `/tmp/tmpfile.txt`. Try to cat that file as bob. While none of the bakers used their discretion to make that recipe available to everyone, it happened anyway as a side effect of the implementation of an application used by the bakers.

This is the nature of discretionary file controls such as those in Unix. They rely on the user knowing what the programs are actually doing. And that is not always the case.

Deliverable#4: (Screenshot) Provide a screenshot of your current terminal.

4.4 Add a privileged user

Add a new user to your system with an id of `lisa`. Lisa will help administer the system and requires the ability to perform privileged functions, sometimes referred to as *sudo*, short for “Superuser do”.

In modern Unix systems, superuser privileges are assigned to members of groups defined in the `/etc/sudoers` file. In older CentOS systems, membership in the group *wheel* provided superuser privileges. That group no longer has special significance. View the `/etc/sudoers` file to see that in our installation, membership in the `admin` group is what provides this privilege. Add lisa to the admin group just as you added mary to the bakers group.

Type the following to see the content of the groups file. Resize the terminal if the full output does not fit the terminal.

```
cat /etc/group
```

Deliverable#5: (Screenshot) Provide a screenshot of your current terminal.

Then login as lisa and perform the privileged function of removing a user. In this case, terminate the bob user with this command:

```
sudo userdel bob
```

Then try to login as bob to confirm the user was deleted.

Type the following to see the current list of users. Resize the terminal if the full output does not fit the terminal.

```
cat /etc/passwd
```

Deliverable#6: (Screenshot) Provide a screenshot of your current terminal.

5 Stopping the lab

After finishing the lab, go to the terminal on your Linux system that was used to start the lab and type:

```
stoplab
```

- (b) **Deliverable#7:** Similar to the previous Unix Lab, you will find a file created for you on the Desktop: `labtainer_xfer`. Go to Desktop → `labtainer_xfer` → `users` and upload that file to Canvas. For example, mine was called `uludag_at_umich.edu.users.lab`. See Figure 1.

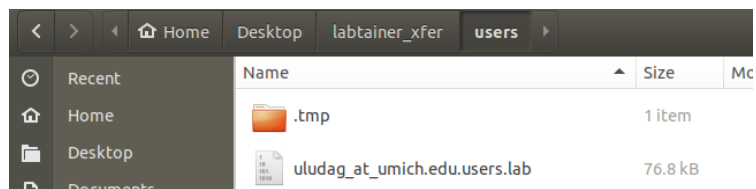


Figure 1: File name to include in the zip file to submit to Canvas for the Users Lab.

Lab #2: Packet Capture Introspection

- (a) Please complete the lab below.

6 Overview

The pcap (packet capture) format is a standard and portable representation of packet-level network traffic. You are likely already familiar with pcap from the previous labs – both Wireshark and tcpdump store and read data in pcap format. This introductory lab is designed to familiarize students with pcaps and traffic analysis using Wireshark. Wireshark includes many powerful tools and is best suited to performing highly targeted analysis on small packet captures. This lab is adapted from [1], which is a helpful resource for improving your familiarity with the Wireshark toolset.

6.1 Background

The student is expected to have at least a basic understanding of the Linux command line, the basics of the file system.

7 Lab Environment

This lab runs in the Labtainer framework, available at <http://nps.edu/web/c3o/labtainers>. That site includes links to a pre-built virtual machine that has Labtainers installed, however Labtainers can be run on any Linux host that supports Docker containers.

From your labtainer-student directory start the lab using:

```
labtainer packet-introspection
```

A link to this lab manual will be displayed.

8 Tasks

8.1 Find Most Active TCP Flow (15 pts)

A common network analysis task is determining the largest contributors to network traffic and potential congestion. In this part you will isolate and examine the largest TCP flow1 in a packet capture. Complete the following steps and answer the questions.

- Open `pcaps/http-misctraffic101.pcapng` in Wireshark
- Select Statistics — Conversations. Click the Ethernet tab; notice there is only one pair of hosts communicating on the local network. Ensure that the “Name resolution” and “limit to display filter” boxes are checked. The MAC address listed as “Cadant” is the local router. The “HewlettP” host is the client from which the traffic was captured.

1. [5 pts] Click on the IPv4 tab to examine the IPv4 conversations in this trace file. Based on the bytes count, what IP addresses participate in the most active IPv4 conversation?
 - Click the TCP tab to identify the most active TCP conversation. Sort by bytes by clicking on the Bytes column heading.
 - Deliverable#8: (Screenshot)** Provide a screenshot of highlighted row for the most active IPv4 conversation.
2. [10 pts] Right-click on the most active TCP conversation and select Apply as Filter → Selected → (A ↔ B). Wireshark automatically creates and applies a display filter for this TCP conversation. How many packets match this filter?
 - Deliverable#9: (Screenshot)** Provide a screenshot of the wireshark showing the number of matched packets..
- Part 1 clean-up: Click the Clear button (the red 'X' next to the filter expression) to remove your display filter before continuing. Toggle to the Conversation window and click Close.

8.2 Geolocating IP Addresses (15 pts)

Correlating network interfaces/IP addresses to physical locations is often a useful task. Wireshark includes a basic capability in this regard, which utilizes the free versions of the MaxMind2 database. It is important to recognize that no IP-geolocation database is error-free.

- Open `pcaps/http-browse101c.pcapng` in Wireshark.
 - Now select Edit → Preferences → Name Resolution4 and click the GeoIP database directories Edit button. Click New and point to the maxmind directory (which has database files downloaded from <http://dev.maxmind.com/geoip/legacy/geolite/>). Continue to click OK until you have closed the GeoIP database paths windows and the Preferences window. Restart Wireshark.
 - Select Statistics → Endpoints and click on the IPv4 tab. You should see information in the Country, City, Latitude, and Longitude columns.
 - Click the Map button. Wireshark will launch a map view in your browser with the known IP addresses plotted as points on the map. Click on any of the plot point to find more information about the IP address.
3. [15 pts] How much aggregate traffic went to/from Milpitas, CA?
 - Deliverable#10: (Screenshot)** Provide a screenshot of the wireshark endpoints window with Country, City, Latitude and Longitude information..

Deliverable#11: (Screenshot) Provide a screenshot of the map for the location showing Milpitas, Ca details.

- Part 2 clean-up: Close the browser tab/window when you are finished. Close the Wireshark Endpoints window

8.3 Reassemble text from TCP stream (15 pts)

As a byte-stream oriented protocol, TCP segments data based on its MSS (Maximum Segment Size), not based on semantics of the English language, or even application data formatting. Thus it can be helpful to reassemble this data before manually inspecting it.

- Open `pcaps/http-wiresharkdownload101.pcapng` in Wireshark.
 - The first three packets are the TCP handshake for the web server connection. Frame 4 contains the client's GET request for the `download.html` page. Right-click on frame 4 and select Follow → TCP stream. Traffic from the first host seen in the trace file, the client in this case, is colored red. Traffic from the second host seen in the trace file, the server in this case, is colored blue.
4. [5 points] Wireshark displays the conversation without the Ethernet, IP or TCP headers. Scroll through the stream to look for the hidden message from Gerald Combs, creator of Wireshark. It is located in the server stream and begins with "X-Slogan". What is the message?

Deliverable#12: (Screenshot) Provide a screenshot of the hidden message in the Follow → TCP Stream window.

- This isn't the only message hidden in the web browsing session. Now that you know the message begins with "X-Slogan", you can have Wireshark display every frame that includes this ASCII string. Click the Close button and then the Clear button to remove the TCP stream filter.
 - Lookup the wireshark manual or do an Internet search to find all the packets containing X-Slogan string in the HTTP payload.
 - Right-click on the other displayed frames and select Follow → TCP Stream to examine the HTTP headers exchanged between hosts. Did you find the other message? Note that you can only follow one stream at a time using this right-click method. You will need to clear out your display filter before following the next stream.
5. [10 pts] What other message did you find (different than Q4)?

Deliverable#13: (Screenshot) Provide a screenshot of the 2nd hidden message in the Follow → TCP Stream window.

- Part 3 clean-up: Click the Close button on the Follow TCP Stream window when you have finished following streams. Click the Clear button to remove your display filter before continuing.

8.4 Extract binary file from FTP session (15 pts)

In the previous section, we extracted ASCII-text messages from packets. What about binary data? Wireshark has tools for this as well.

- Open `pcaps/ftp-clientside101.pcapng` in Wireshark.
- Scroll through the beginning of the trace file. You will see numerous FTP commands used to login, request a directory, define a port number for the data transfer and retrieve a file.

- There are two data connection in this trace file: one for the directory list and another for the file transfer. We are only interested in these two data streams, and not the command channel stream. In the Follow → TCP Stream window, click the **Hide This Stream** button. This closes the TCP stream window and applies an exclusion filter.
6. [5 pts] Now you only see the data channel traffic. Frames 16 through 18 and frames 35 through 38 are TCP handshake packets to establish the two required data channels. Right-click on frame 16 and select Follow → TCP Stream. This stream list indicates there is only one file in the directory. What is its name? (You will use it next.)
- Deliverable#14: (Screenshot)** Provide a screenshot showing the file name in the Follow → TCP Stream window.
- Click the Filter Out This Stream button. This closes the *TCP stream* window and adds to the existing exclusion filter.
 - The only remaining traffic displayed is the file transfer traffic (FTP-DATA). Right-click on any frame and select Follow → TCP Stream. You can view the file identifier that indicates this is a .jpg file (JFIF) and the metadata contained in the graphic file.
 - To reassemble the graphic image transferred in this FTP communication, Change the Show and save data as dropdown to “Raw” and click the Save As button, select a target directory for the file, and set the file name to the one you found a few steps back. Click Save.
7. [10 pts] Navigate to the target directory and open the file you saved in the previous step.
- Deliverable#15: (Screenshot)** Provide a screenshot of the file you saved in the previous step.

You can save this file to your host machine in two different ways:

- Within the lab terminal you ran wireshark commands above: `scp jyour file name will go herej student@172.17.0.1:.`
The password is **password123**. The file will be saved in your VM's /home/ubuntu directory.
- You can copy the file to the host machine by using the following command on the terminal (student@ubuntu: /labtainer/labtainer-student\$ prompt) you are using to run the **labtainer** and **stoplab** commands:

```
sudo docker cp packet-introspection.ws.student:/home/ubuntu/jyour file name will go herej
```

The password is **password123**.

- Part 4 clean-up: When you've finished examining the image you extracted, close your image viewer. Return to Wireshark to close the TCP Stream window and clear your display filter

9 Submission

After finishing the lab, go to the terminal on your Linux system that was used to start the lab and type:

```
stoplab
```

- (b) **Deliverable#16:** Similar to the previous Unix Lab, you will find a file created for you on the Desktop: `labtainer_xfer`. Go to Desktop → `labtainer_xfer` → `packet-introspection` and upload that file to Canvas. For example, mine was called `uludag_at_umich.edu.packet-introspection.lab`. See Figure 2.

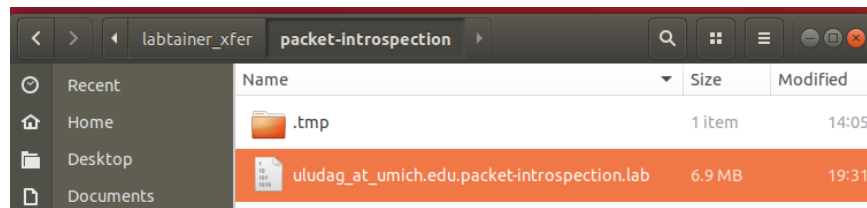


Figure 2: File name to include in the zip file to submit to Canvas for the packet-introspection Lab.

10 References

- [1] Wireshark 101: Essential Skills for Network Analysis, by Laura Chappell and Gerald Combs. Published by Protocol Analysis Institute, 2013. ISBN: 1893939723, 9781893939721

A Preparing the Lab Infrastructure

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

1. Download and install Virtual Box for your computer's OS: <https://www.virtualbox.org/wiki/Downloads>
2. Download VirtualBox VM Appliance : [VirtualBox VM Appliance](#). 3.6GB! It will take a while to download depending upon your Internetn connection speed.
3. Start VirtualBox, and then: File / Import Appliance / LabtainerVM-VirutalBox.ova, Next, Import. Might take a couple of minutes.
4. Once the importing of the image is finished, click on the green left arrow to start the appliance. It will take a while for it to start and to settle. Please there are many keyboard shortcuts for VirtualBox, The Host key is defined to be the right CTRL key. For example, Host+F (Right CTRL + F) to get in and out of Full Screen Mode. Figure 3 shows a screenshot of the intial screen after the startup.

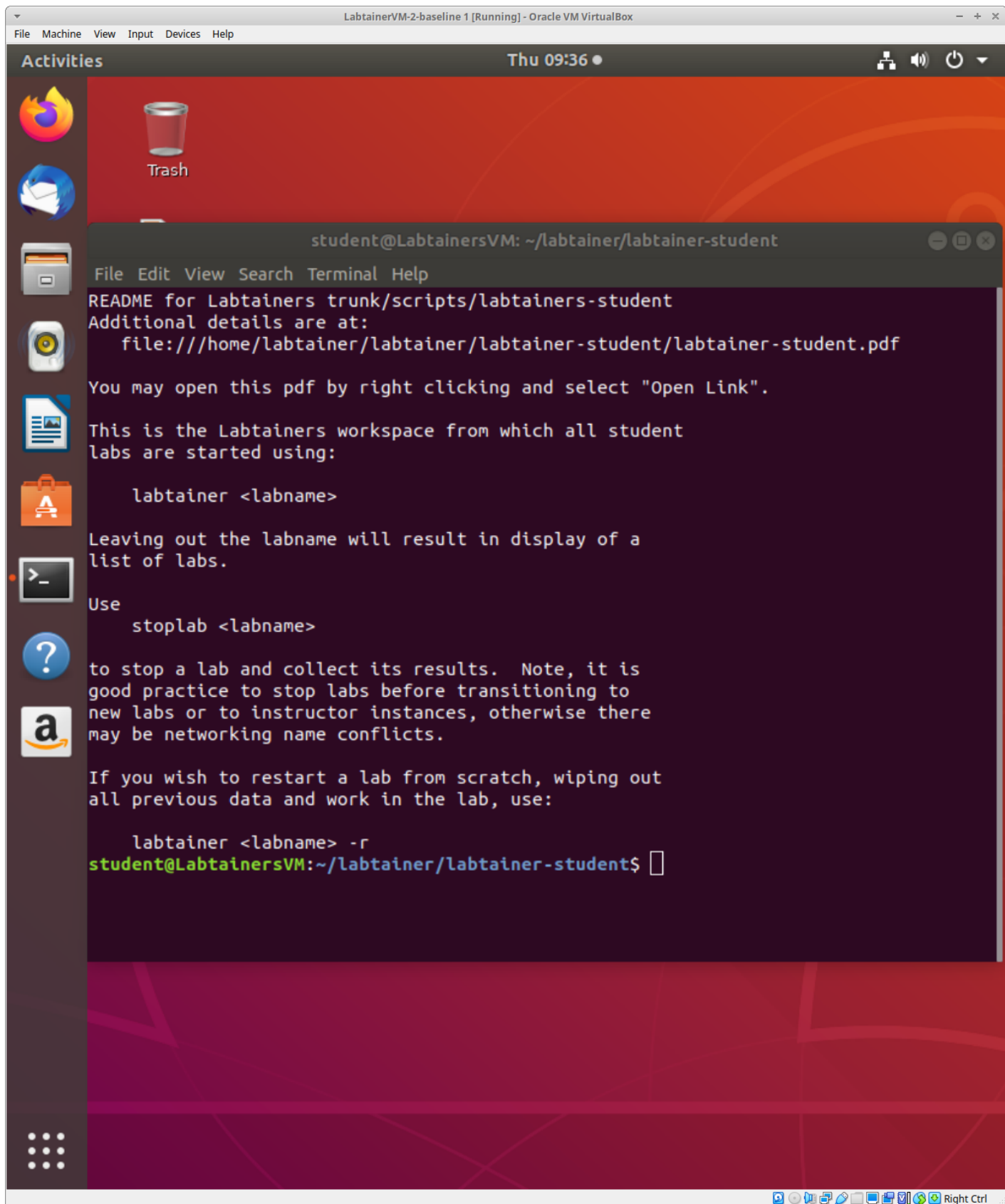
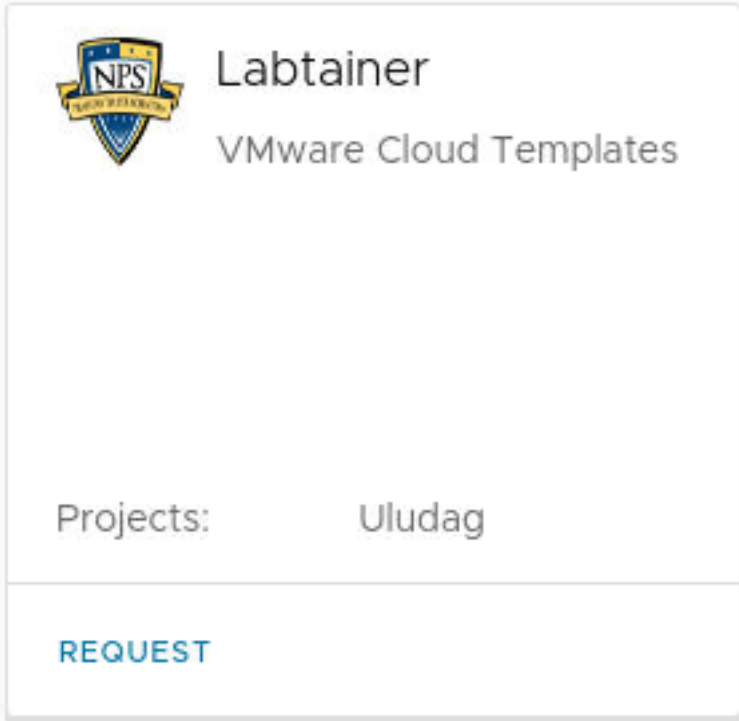


Figure 3: After starting the VirtualBox VM Appliance. Initial screen.

Preparing the Lab infrastructure - Option #2 : To Run on UM-Flint 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.



The screenshot shows a web form for requesting a Labtainer. At the top left is the NPS (National Park Service) logo. To its right, the text "Labtainer" is displayed in a large font, with "VMware Cloud Templates" in a smaller font below it. Further down, there is a label "Projects:" followed by a text input field containing the word "Uludag". At the bottom of the form is a blue button with the word "REQUEST" in white capital letters.

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>

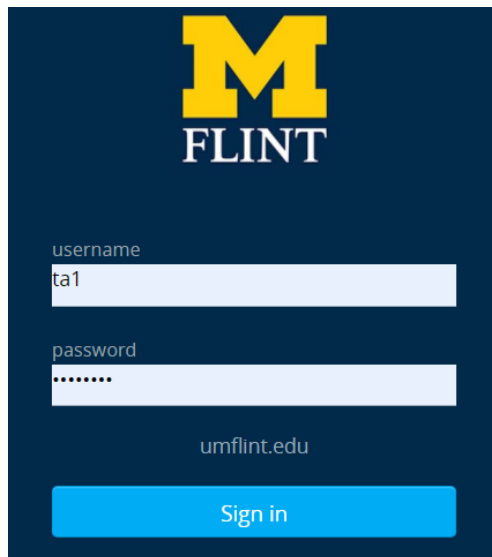
If you are working remotely, establish a VPN connection the university to continue with the rest of this guide.

Login

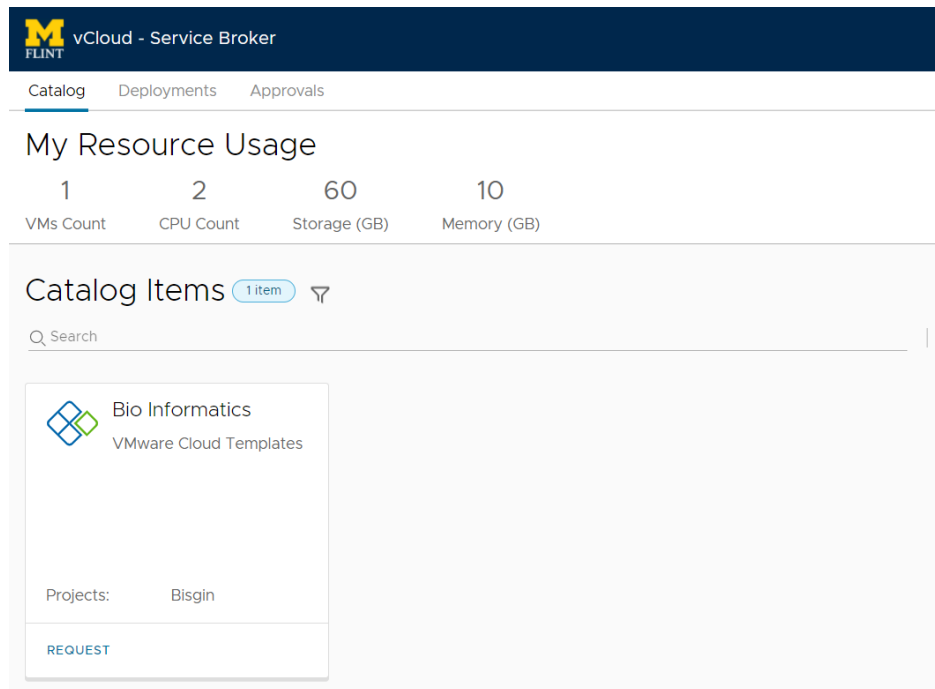
Go to: <https://vcloud.umflint.edu/catalog/#/library>

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.

A screenshot of the UM Flint login portal. At the top is the UM Flint logo, a yellow 'M' above the word 'FLINT' in white. Below the logo are two input fields: 'username' with the text 'ta1' and 'password' with masked characters '.....'. Under the password field is the text 'umflint.edu'. At the bottom is a blue button with the text 'Sign in'.

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.



Deploy

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

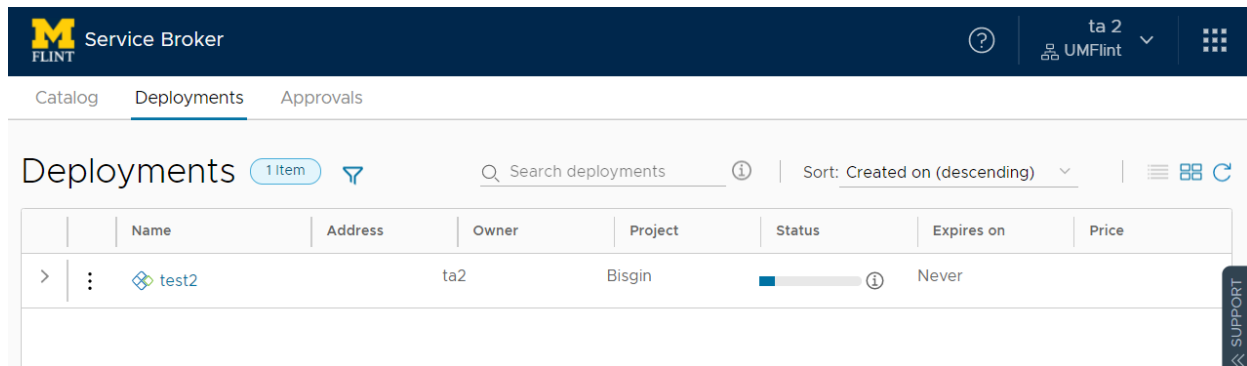
The screenshot shows the 'vCloud - Service Broker' interface with the 'New Request' form. The header is the same as the previous screenshot. The 'Catalog' tab is selected. The form title is 'New Request'. It shows the 'Bio Informatics' logo and the text 'Version bio_vm'. Below this, there are two dropdown menus: 'Project' and 'Deployment Name'. Both have red asterisks indicating they are required fields. At the bottom, there are two buttons: 'SUBMIT' and 'CANCEL'.

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

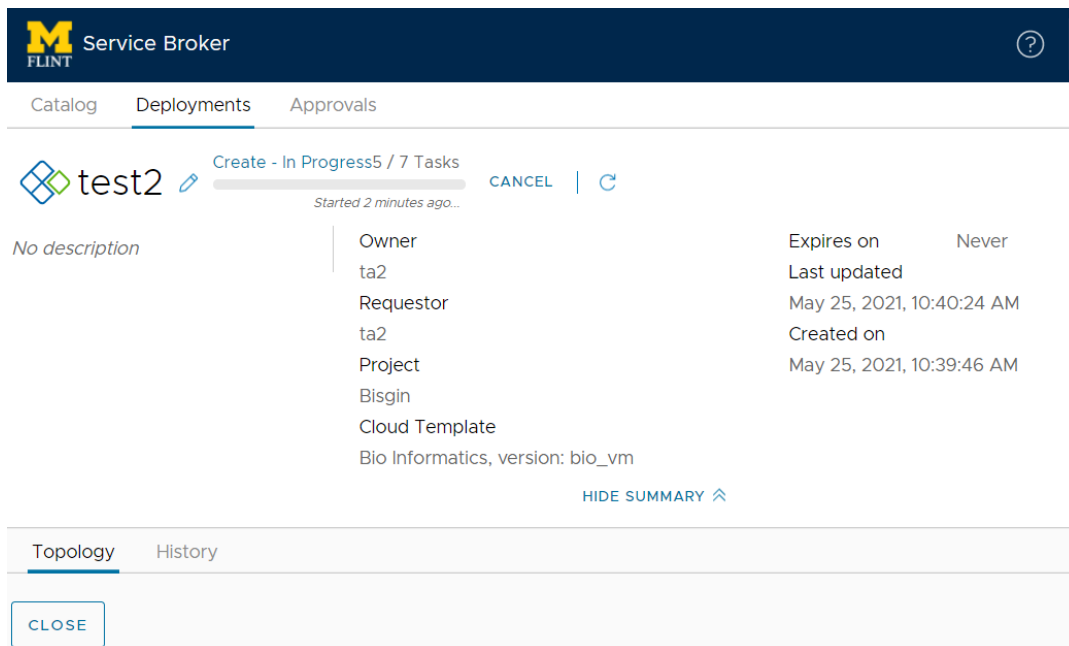
Select the instructor name under the “Project” drop down menu.

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”.



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.



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

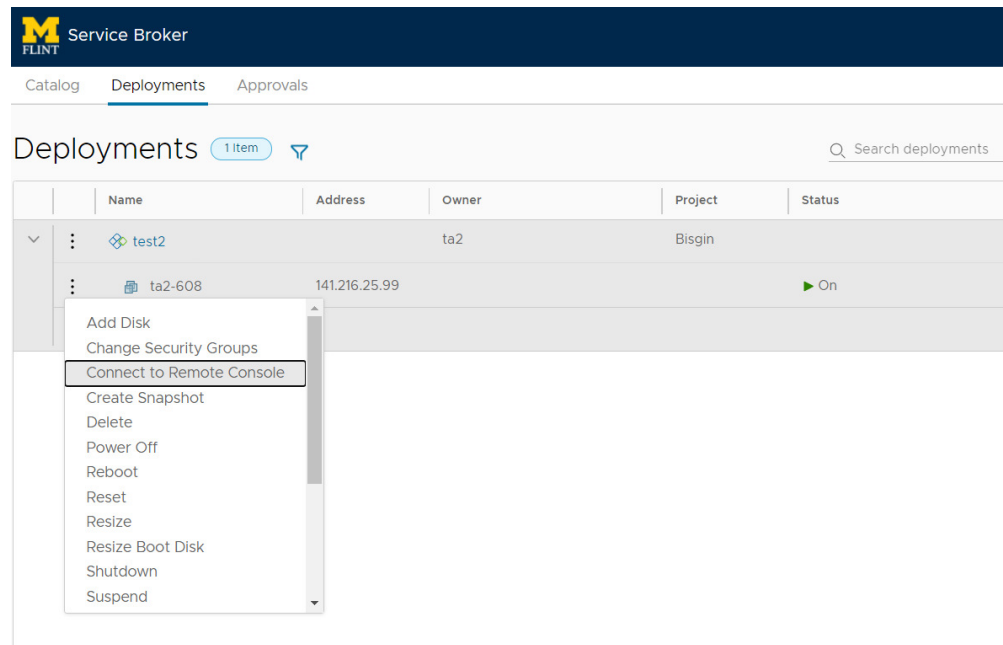
Once a deployment has finished, there are two primary ways you can interact with your VM. Once access is configured, you will probably interact with your VM most often through the use of an integrated or third party connection tool. Depending on the operating system and how your instructor has configured 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.

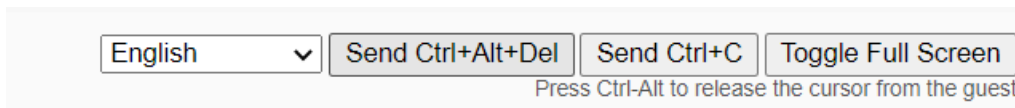
Most VMs will power on once they have deployed. In the “Deployments” section expand a deployment using the arrow to the left of the deployment you want to interact with, you will see the network connection and the VM listed in the deployment.

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.



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 4.

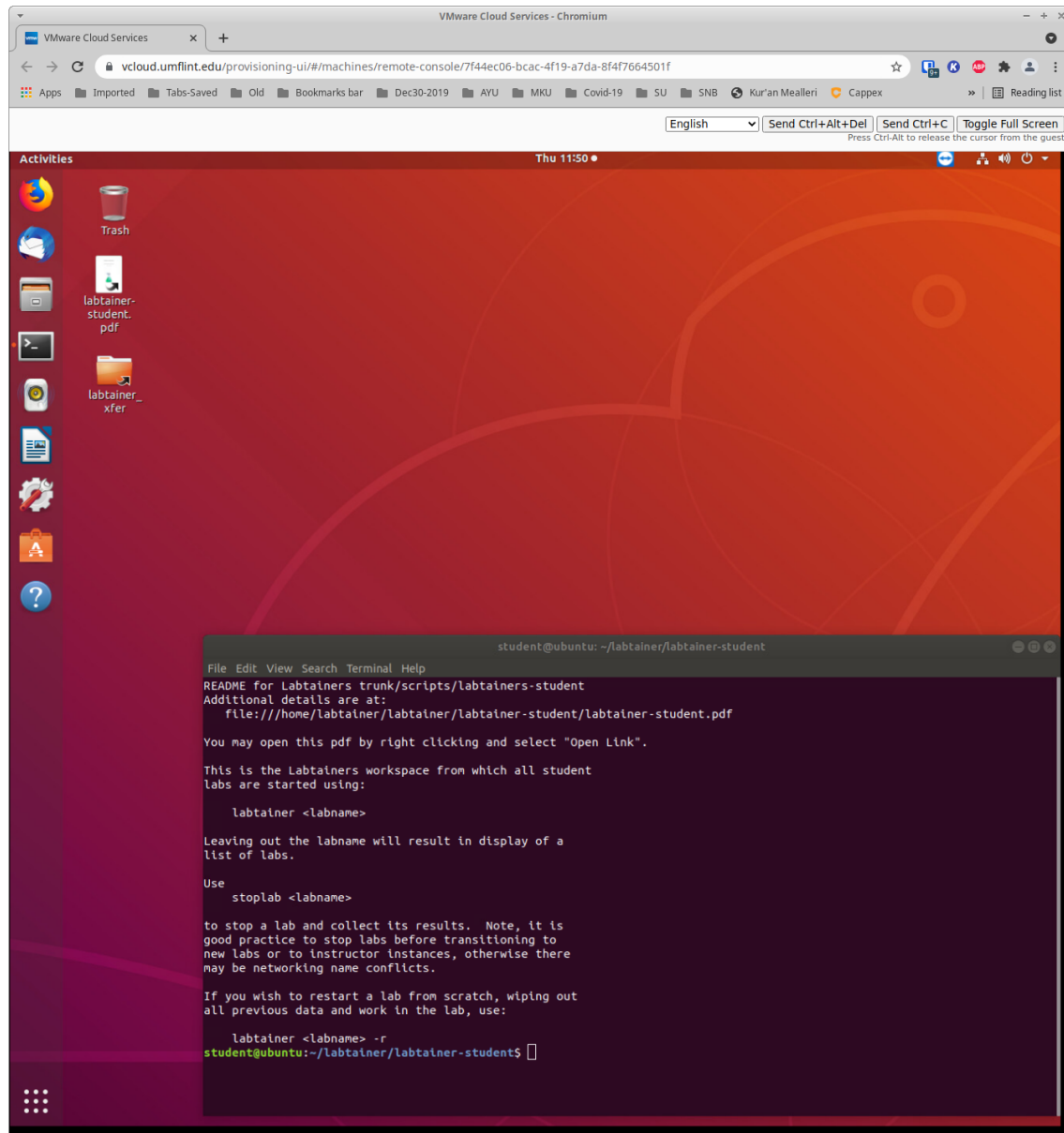


Figure 4: vCloud access to labtainers VM from within a browser (Chromium) over VPN.

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 5 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.

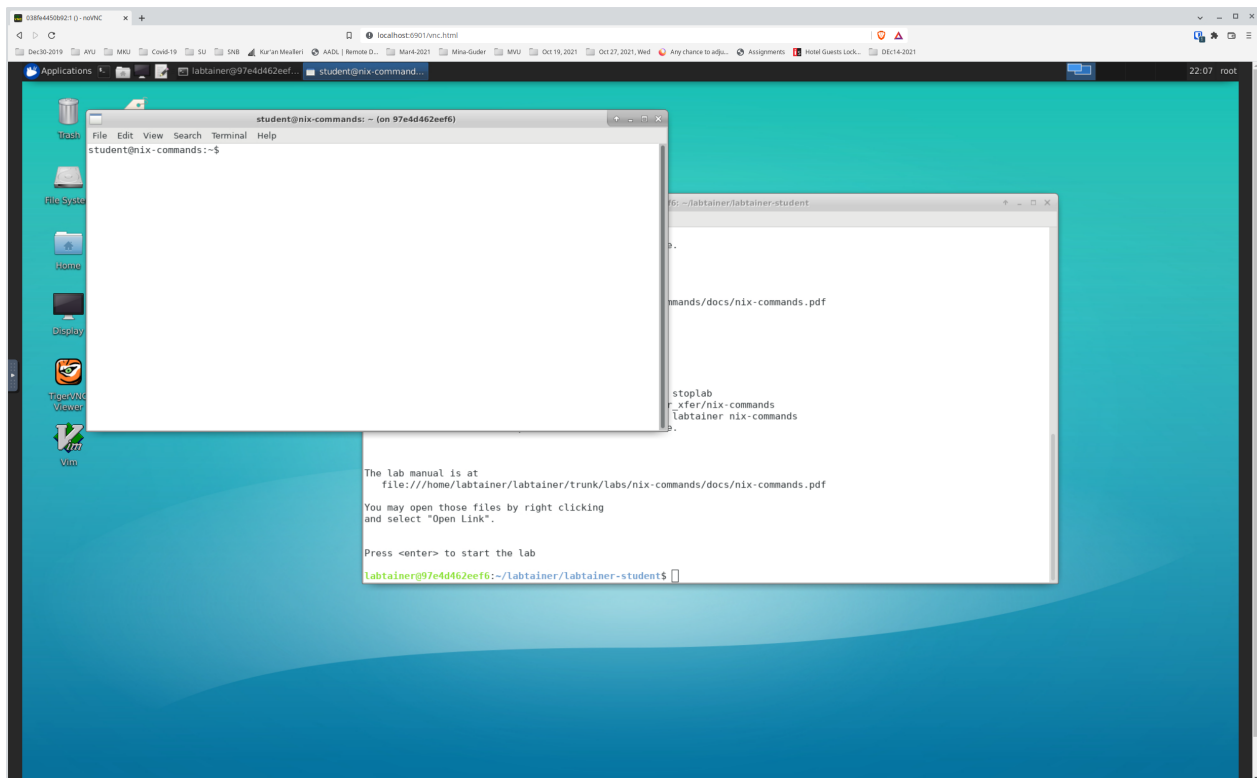


Figure 5: Cloud access for labs using the Azure free allocation from within your browser.

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 passphrase, 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, simply 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 deleted.

- 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.