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 `passwrd` 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 at 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 creates 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.

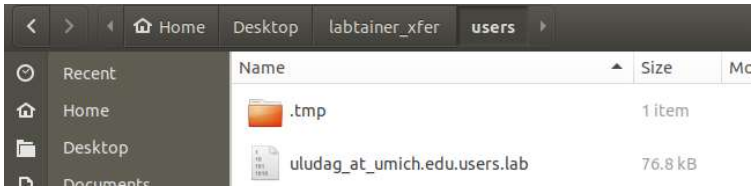

Figure 1: File name to include in the zip file to submit to Canvas for the Users Lab.

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:

   i. Within the lab terminal you ran wireshark commands above: scp ¡your file name will go here¿ student@172.17.0.1:.
   The password is password123. The file will be saved in your VM's /home/ubuntu directory.

   ii. 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/¡your file name will go here¿ .
   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.

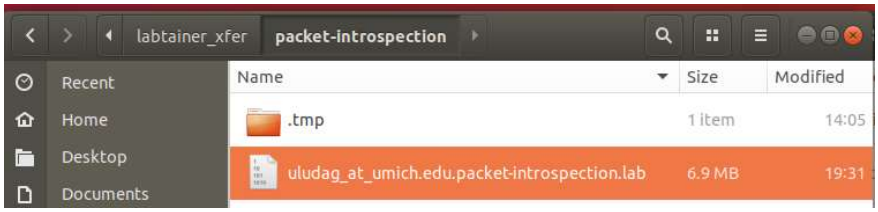

Figure 2: File name to include in the zip file to submit to Canvas for the packet-introspection Lab.

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