Lab 2.2

By: Nathan Metens (metens@pdx.edu)

Task 1: Accounts - useradd, passwd, id 1
Task 1.1: Duplicate UIDs 2
Task 2: umask 3
Task 3: chmod 4
Task 4: groups, newgrp, chgrp 4
Task 5: setuid bit, sudo 5
Task 6: Using sudo, chown 5
Task 7: Process - w, tty, /proc 9
Task 8: pstree, killall 11
Task 9: Configuration issues - PATH hijacking 12
Task 10: sshd 15
Task 11: TryHackMe Linux process analysis 16

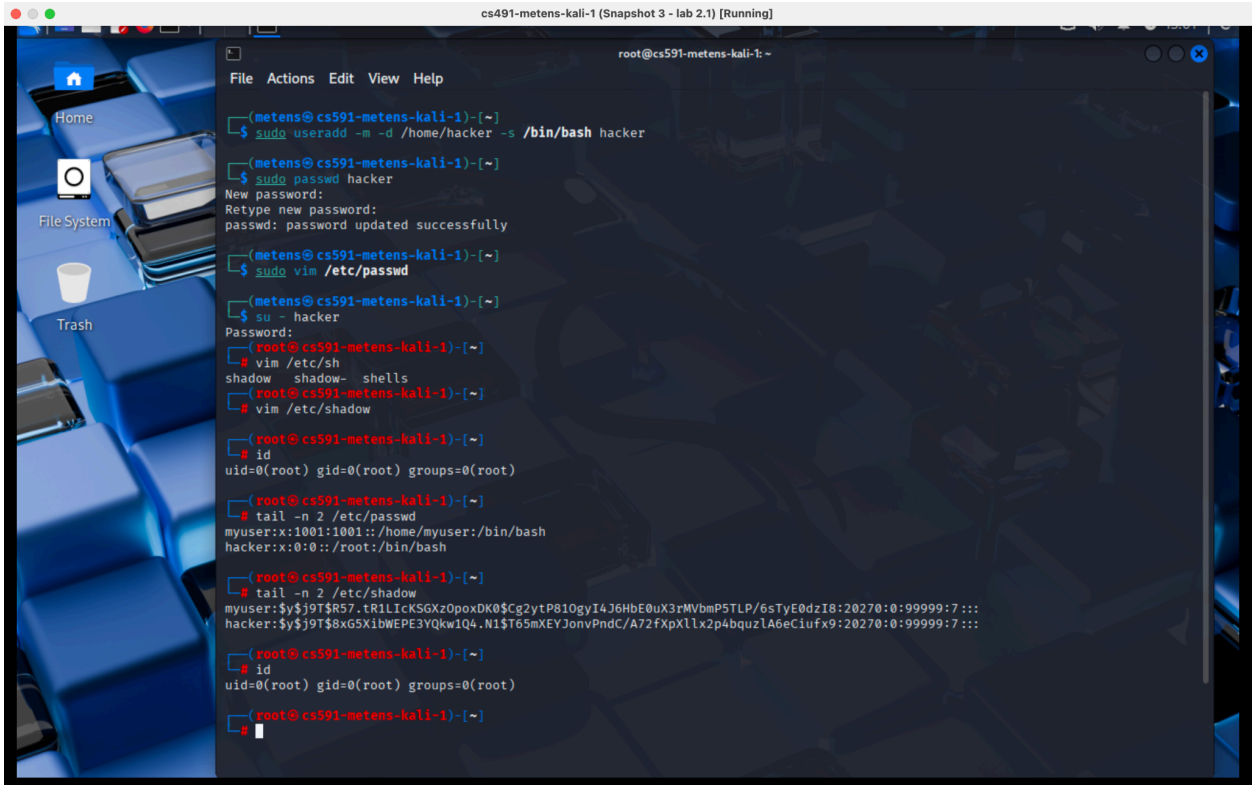
Task 1: Accounts - useradd, passwd, id

The uid and gid of my account (metens) are 1000 and 1000, respectively.
The uid and gid of the new account (myuser) are 1001 and 1001, respectively.

Task 1.1: Duplicate UIDs

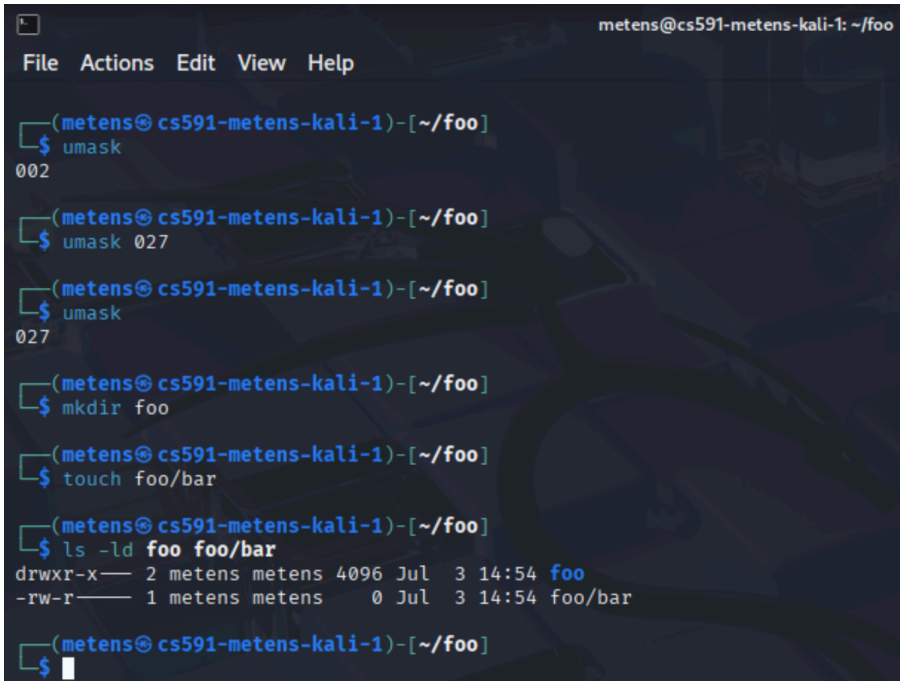
```
metens:x:1000:1000:metens,,,:/home/metens:/usr/bin/zsh
myuser:x:1001:1001::/home/myuser:/bin/bash
hacker:x:0:0::/root:/bin/bash
~
"/etc/passwd" 57L, 3226B written
```

```
metens:$y$j9T$I7.y2OWFBwuV93.0TH7dN.$YOaq2pn3LRpsoQujArVGsxdgp06rSOQCaIg8KXz.IY6:20265:0:99999:7:::
myuser:$y$j9T$R57.tR1LIcKSGXzOpoxDK0$Cg2ytP81OgyI4J6HbE0uX3rMVbmP5TLP/6sTyE0dzI8:20270:0:99999:7:::
hacker:$y$j9T$8xG5XibWEPE3YQkw1Q4.N1$T65mXEYJonvPndC/A72fXpXllx2p4bquzlA6eCiufx9:20270:0:99999:7:::
```



Task 2: umask

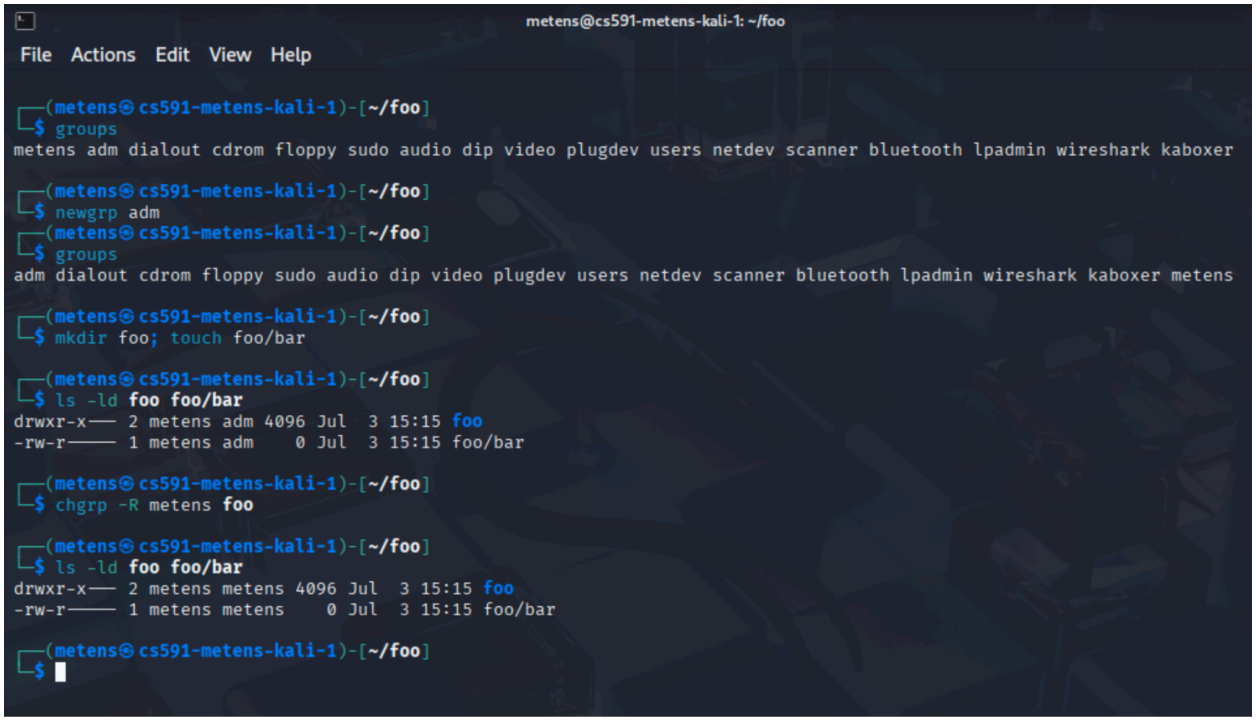
Since file permissions are organized in an owner-group-others fashion, and each group can have a bit for read, write, and execute, this means that for full access, rwx -> 111, the first value in the umask (owner) must be 7. Then, for read and execute permissions on the group, r-x -> 101, the second umask value will be 5. Finally, no access to others would be 0. So we have 750 as our file permissions. To get the umask value, we subtract out permissions from the maximum value for full permissions, which would be 777. So we get a umaks of 777-750 = **027**. Now we can change to that umask value in the shell:



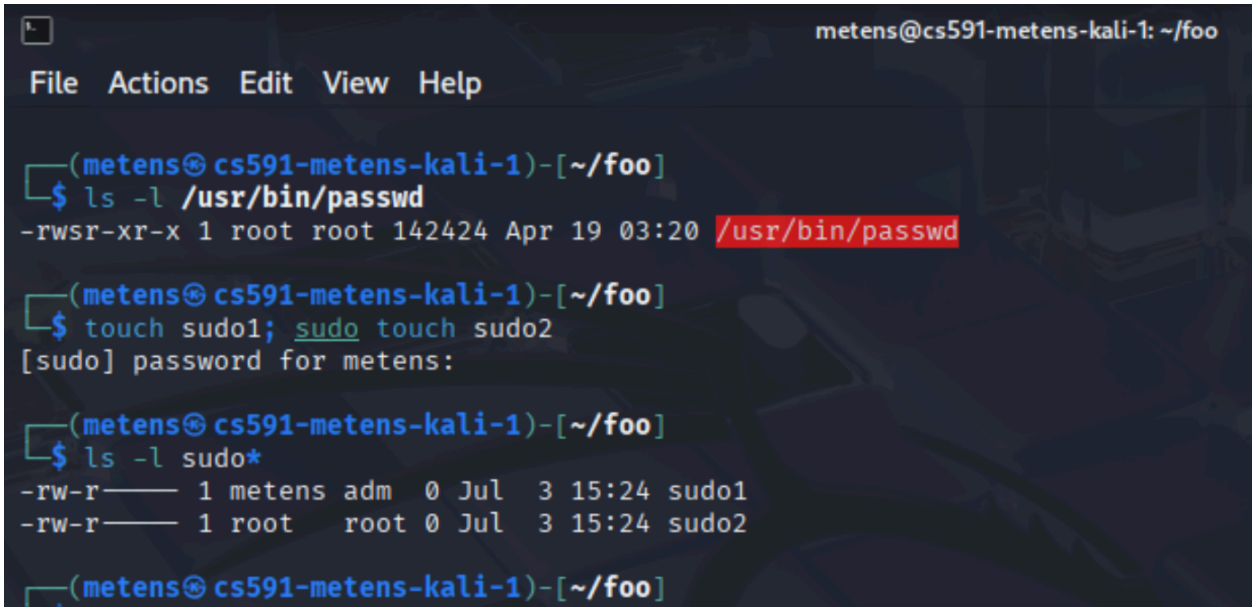
Task 3: chmod

I used `-R` for the recursive flag, ‘o’ for others, and a ‘+rX’ for updating the permissions. The lowercase r for reading the foo/bar file, and the uppercase X for executing the dir foo.

Task 4: groups, newgrp, chgrp

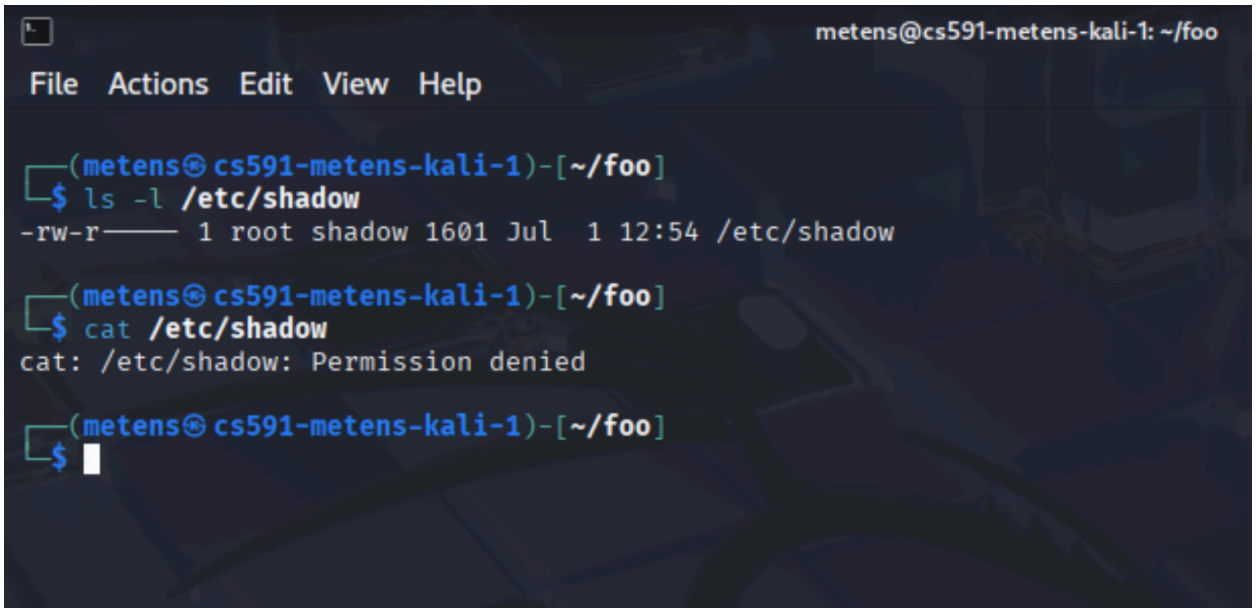


Task 5: setuid bit, sudo



It appears that both files have equal permissions for the owner. However, for one, the owner is metens, and the other's owner is root. I, metens, will be able to read and write the contents of the file, and so will root, but others can't do anything to it, and groups can only read the file.

Task 6: Using sudo, chown



Since the shadow file has no read, write, or execute permissions for the others group, this means I am unable to cat the data inside the file.

```
┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ sudo cat /etc/shadow | egrep "myuser"
myuser:$y$j9T$R57.tR1LIcKSGXzOpoxDK0$Cg2ytP81OgyI4J6HbE0uX3rMVbmP5TLP/6sTyE0dzI8:20270:0:99999:7:::
```

```
┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ mkdir foo; touch foo/bar

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ sudo chown -R myuser foo

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ ls -ld foo foo/bar
drwxr-x--- 2 myuser adm 4096 Jul  3 16:16 foo
-rw-r----- 1 myuser adm    0 Jul  3 16:16 foo/bar

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$
```

To remove the file and dir, I was not the owner, so I changed to the myuser. However, the contents weren't in myuser's dir. So I tried to delete them from myuser by accessing metens dir structure. This was not possible:

```
┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ ls -ld foo foo/bar
drwxr-x--- 2 myuser adm 4096 Jul  3 16:16 foo
-rw-r----- 1 myuser adm    0 Jul  3 16:16 foo/bar

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ rm -r *
zsh: sure you want to delete the only file in /home/metens/foo [yn]? y
rm: descend into write-protected directory 'foo'? y
rm: remove write-protected regular empty file 'foo/bar'? y
rm: cannot remove 'foo/bar': Permission denied

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ su - myuser
Password:
┌──(myuser㉿cs591-metens-kali-1)-[~]
└─$ rm -rf /home/metens/foo
rm: cannot remove '/home/metens/foo': Permission denied

┌──(myuser㉿cs591-metens-kali-1)-[~]
└─$
```

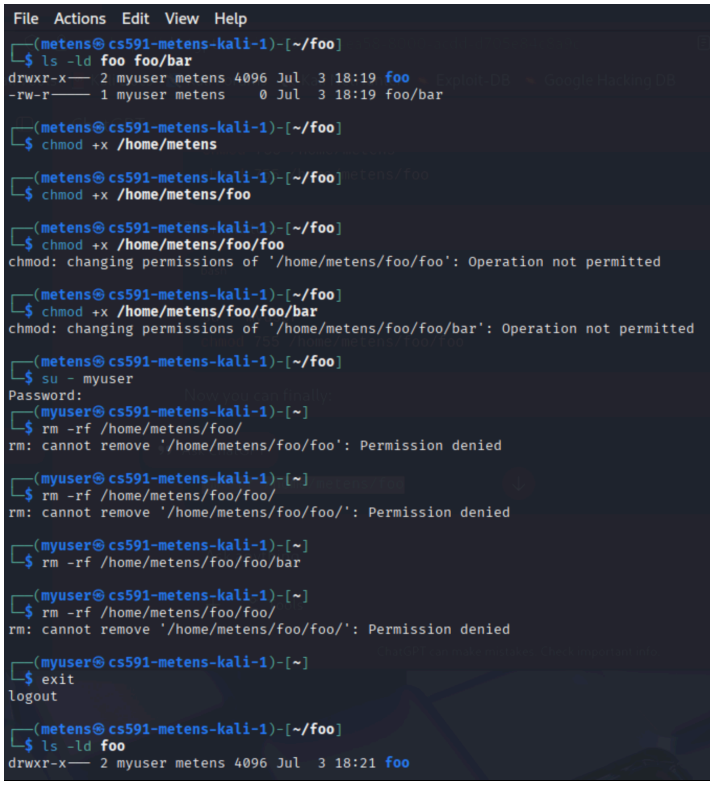
So, I changed back to metens, changed the permissions of the foo dir to be executable by all, and tried to delete from myuser. This also didn't work:

```
┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ chmod +x /home/metens/foo/

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ su - myuser
Password:
┌──(myuser㉿cs591-metens-kali-1)-[~]
└─$ rm -rf /home/metens/foo/
rm: cannot remove '/home/metens/foo/': Permission denied

┌──(myuser㉿cs591-metens-kali-1)-[~]
└─$ exit
logout

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ chmod -R +x /home/metens/foo
chmod: changing permissions of '/home/metens/foo/foo': Operation not permitted
chmod: changing permissions of '/home/metens/foo/foo/bar': Operation not permitted
```

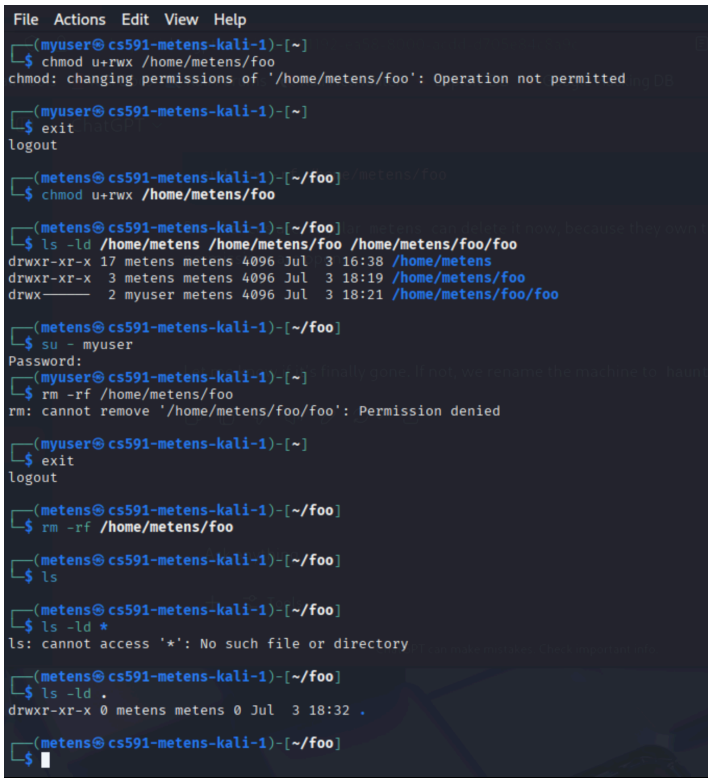
Frustratingly, I couldn't seem to figure this out in less than 30 minutes, which is too long, so I took a break and then searched the web for some guidance… After trying again, I set everything up the same way as before:

```
┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ ls -ld foo foo/bar
drwxr-x--- 2 myuser metens 4096 Jul  3 16:50 foo
-rw-r----- 1 myuser metens    0 Jul  3 16:50 foo/bar

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ rm foo/bar
rm: remove write-protected regular empty file 'foo/bar'? y
rm: cannot remove 'foo/bar': Permission denied

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ rm foo
rm: cannot remove 'foo': Is a directory

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ rm -r foo
rm: descend into write-protected directory 'foo'? y
rm: remove write-protected regular empty file 'foo/bar'? y
rm: cannot remove 'foo/bar': Permission denied

┌──(metens㉿cs591-metens-kali-1)-[~/foo]
└─$ ls -ld foo foo/bar
drwxr-x--- 2 myuser metens 4096 Jul  3 16:50 foo
-rw-r----- 1 myuser metens    0 Jul  3 16:50 foo/bar
```

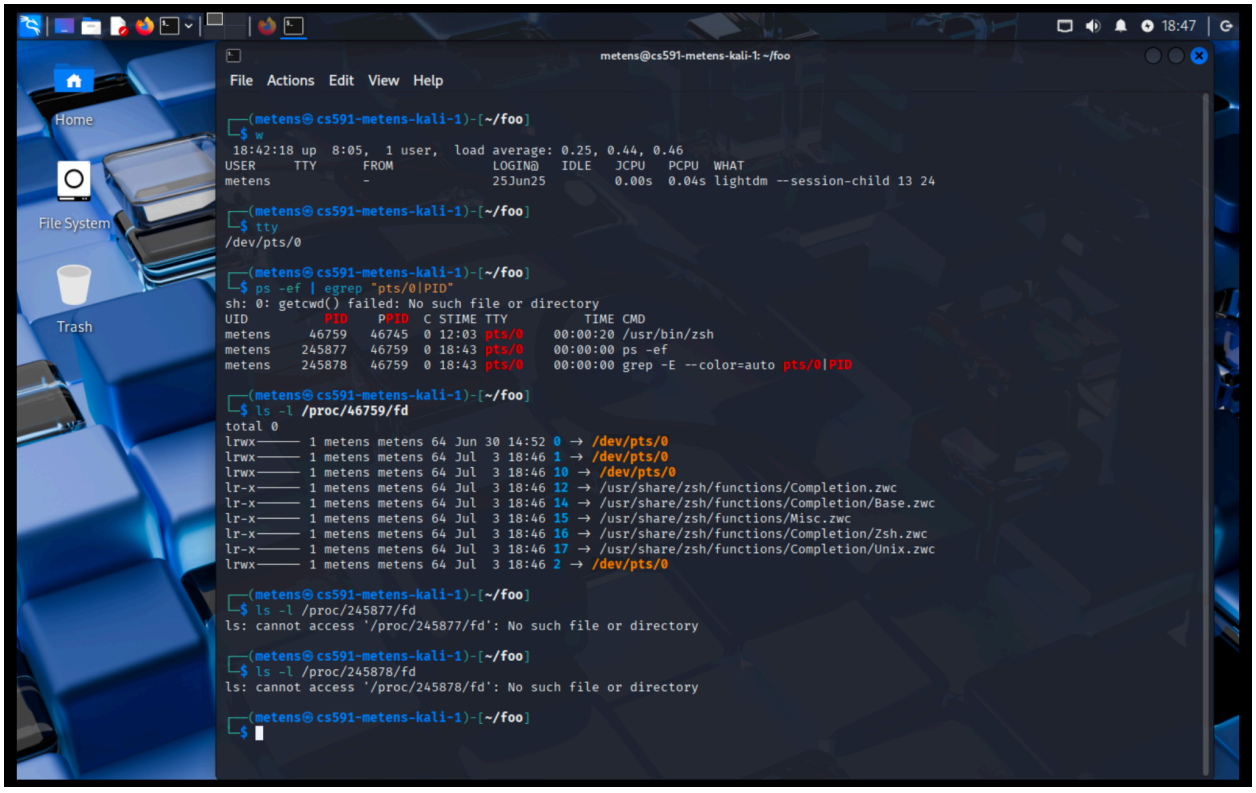
Here is what I did to be able to delete the bar file:



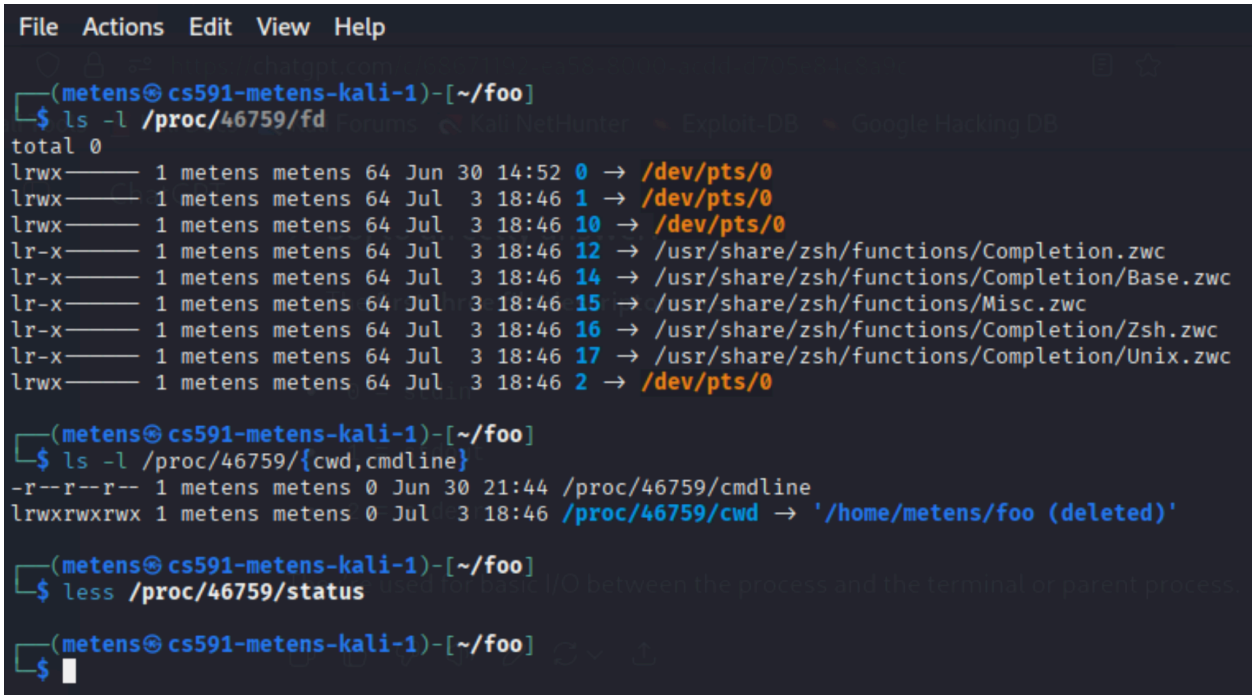
After another 20 minutes of testing and changing mods around, I did this, and it magically worked:

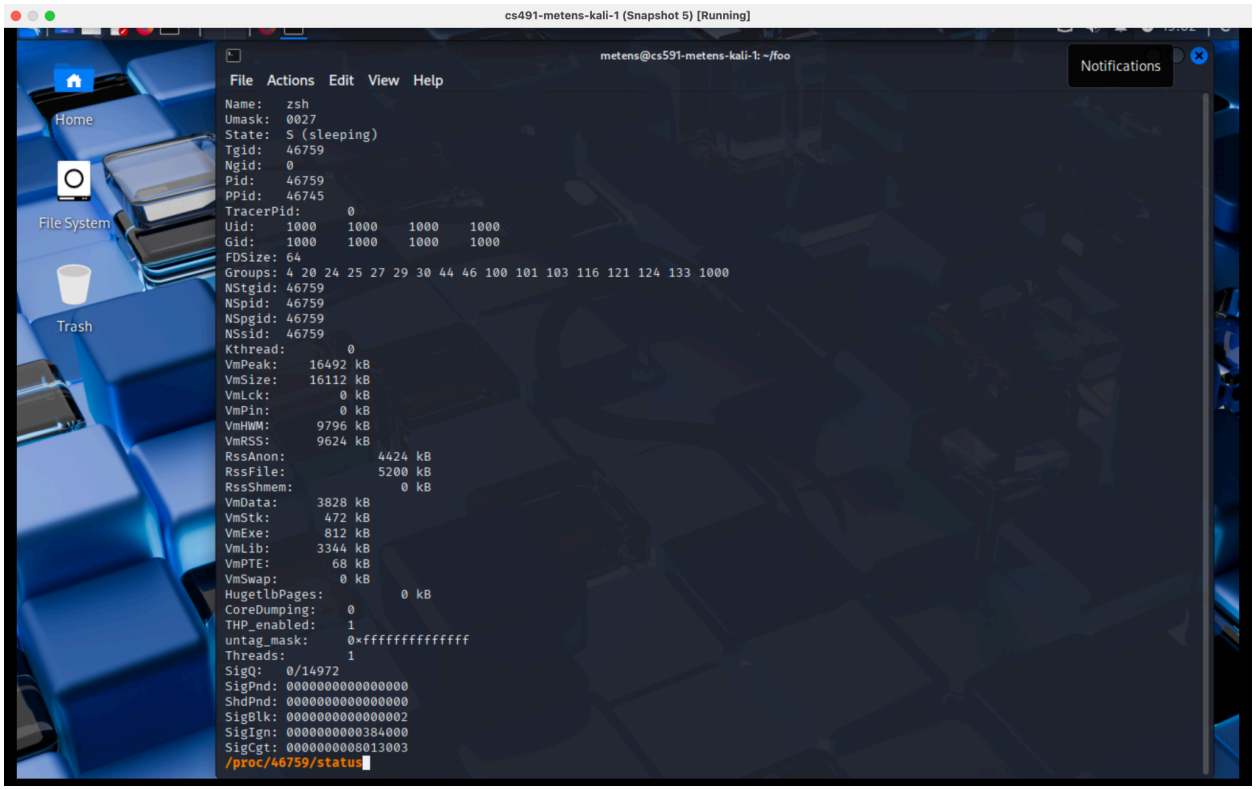


Task 7: Process - w, tty, /proc



After doing some research, the 0 -> /dev/pts/0, 1 -> /dev/pts/0, and 2 -> /dev/pts/0 are linux processes. The 0 references stdin (read), 1 references stdout (write), and 2 references stderr (write and errors). These processes are used for input and output operations on the terminal.





The line in the status output that tells us that any file created will be readable by anyone is the line containing “Umask: 0027”. If we do 777-27, we get 750, which means that the owner can rwx, the groups can r-x, but the others cannot do anything 0 -> —.

Task 8: pstree, killall

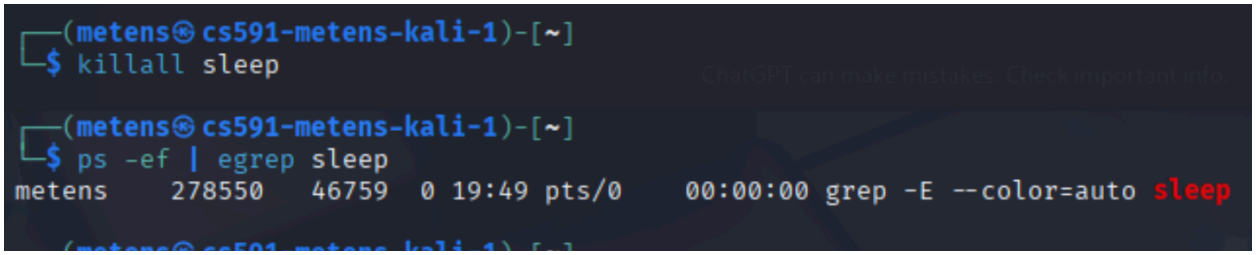
```
File  Actions  Edit  View  Help

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$ vim sleep.bash

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$ chmod +x sleep.bash

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$ ./sleep.bash &
[1] 269114

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$ ps -ef | egrep 269114
metens     269114   46759  0 19:30 pts/0    00:00:00 /bin/bash ./sleep.bash
metens     269116  269114  0 19:30 pts/0    00:00:00 sleep 1000
metens     269117  269114  0 19:30 pts/0    00:00:00 sleep 2000
metens     269118  269114  0 19:30 pts/0    00:00:00 sleep 3000
metens     269119  269114  0 19:30 pts/0    00:00:00 sleep 4000
metens     269245   46759  0 19:30 pts/0    00:00:00 grep -E --color=auto 269114

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$
```

The 4 sleep commands pop up in the ps listing because each process is independent of the others. The first process (running the bash script in the background), 269114, is the parent process to all the other processes. Each process is run in the background, “hence the & in the script”. The parent id 269114 corresponds to the `./sleep.bash` process that is also in the background because we performed a `./sleep.bash &` to put it in the background as well.

```
        ├─obexd───4*[{obexd}]
        ├─pipewire───2*[{pipewire}]
        ├─pipewire───{pipewire}
        ├─pipewire-pulse───2*[{pipewire-pulse}]
        ├─wireplumber───5*[{wireplumber}]
        ├─xdg-desktop-por───5*[{xdg-desktop-por}]
        ├─xdg-desktop-por───4*[{xdg-desktop-por}]
        ├─xdg-document-po─┬─fusermount3
        │                 └─6*[{xdg-document-po}]
        ├─xdg-permission-───3*[{xdg-permission-}]
        └─xfce4-notifyd───4*[{xfce4-notifyd}]
    ├─systemd-journal
    ├─systemd-logind
    ├─systemd-timesyn───{systemd-timesyn}
    ├─systemd-udevd
    ├─udisksd───5*[{udisksd}]
    ├─upowerd───3*[{upowerd}]
    └─xcape───{xcape}

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$ pstree | grep "sleep"
      |           |     `-sleep.bash───4*[sleep]
      ├─3*[sleep]

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$ kill 269114
```

```
        ├─xdg-permission-───3*[{xdg-permission-}]
        └─xfce4-notifyd───4*[{xfce4-notifyd}]
    ├─systemd-journal
    ├─systemd-logind
    ├─systemd-timesyn───{systemd-timesyn}
    ├─systemd-udevd
    ├─udisksd───5*[{udisksd}]
    ├─upowerd───3*[{upowerd}]
    └─xcape───{xcape}

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$ pstree | grep "sleep"
      ├─7*[sleep]

┌──(metens㉿cs591-metens-kali-1)-[~]
└─$ ps -ef | egrep sleep
metens     268445       1  0 19:29 pts/0    00:00:00 sleep 1000
metens     268446       1  0 19:29 pts/0    00:00:00 sleep 2000
metens     268447       1  0 19:29 pts/0    00:00:00 sleep 3000
metens     269116       1  0 19:30 pts/0    00:00:00 sleep 1000
metens     269117       1  0 19:30 pts/0    00:00:00 sleep 2000
metens     269118       1  0 19:30 pts/0    00:00:00 sleep 3000
metens     269119       1  0 19:30 pts/0    00:00:00 sleep 4000
metens     274118   46759  0 19:40 pts/0    00:00:00 grep -E --color=auto sleep
```

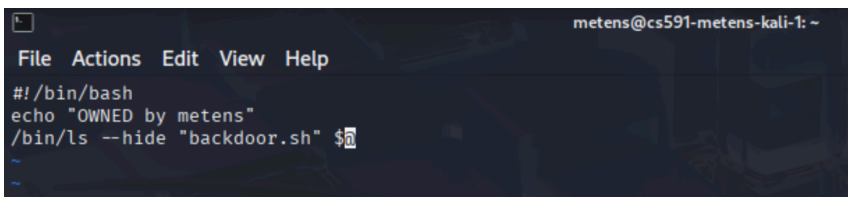
After killing the `./sleep.bash` parent process by its PID, each other sleep process built off of the first one. I did this twice, so there are duplicate processes. However, the first process of sleep 1000 has a PID of 269116, and the other 3 have one count higher than that PID each time. All of them seem to be running independently of each other, so there is no parent process in this case. So, their PPIDs correspond to themselves.



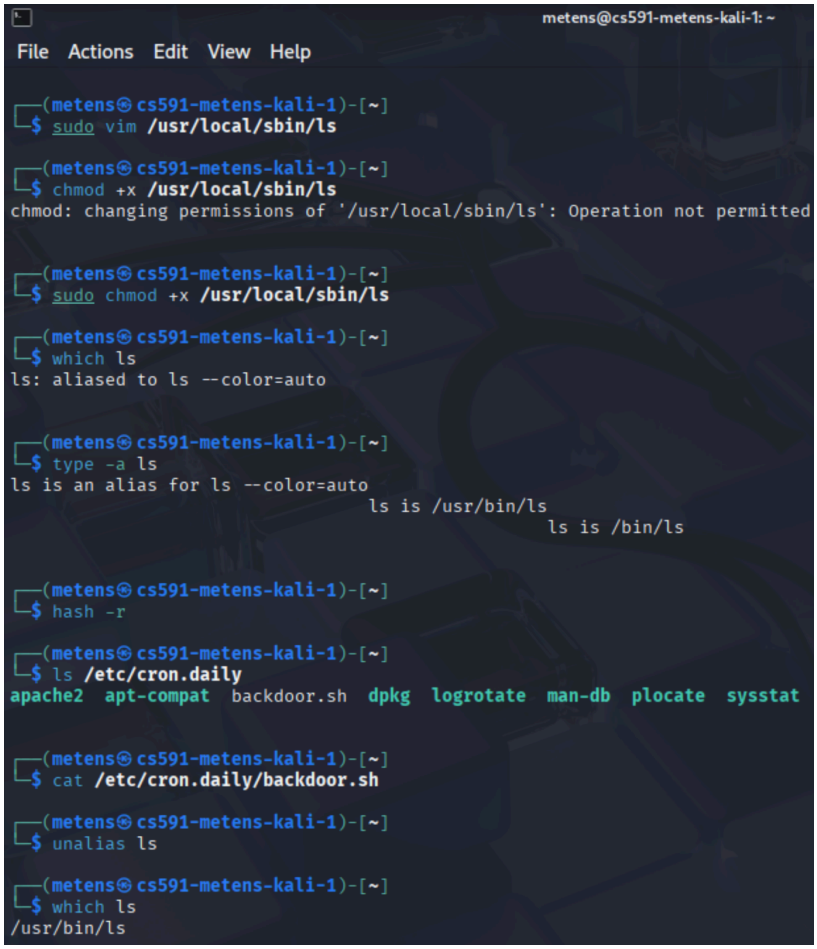
Task 9: Configuration issues - PATH hijacking

Since the `$PATH` is `/home/metens/.local/bin:/usr/local/sbin:...` and it searches each dir in order from left to right, it will come across the `/bin/ls` directory first before the `/usr/bin/ls` directory to find the correct ls for listing stuff.

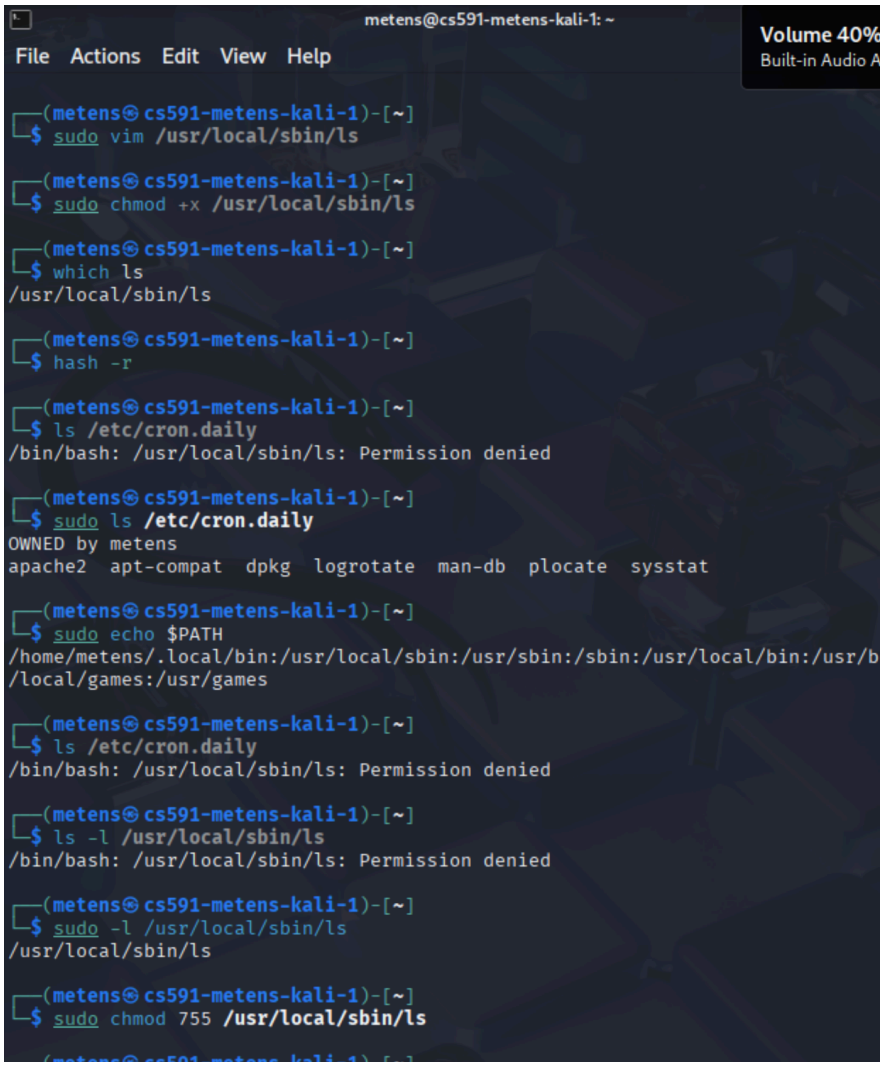
As we can see from the `file /bin/ls` command, the ls command is an “ELF executable” binary file type.



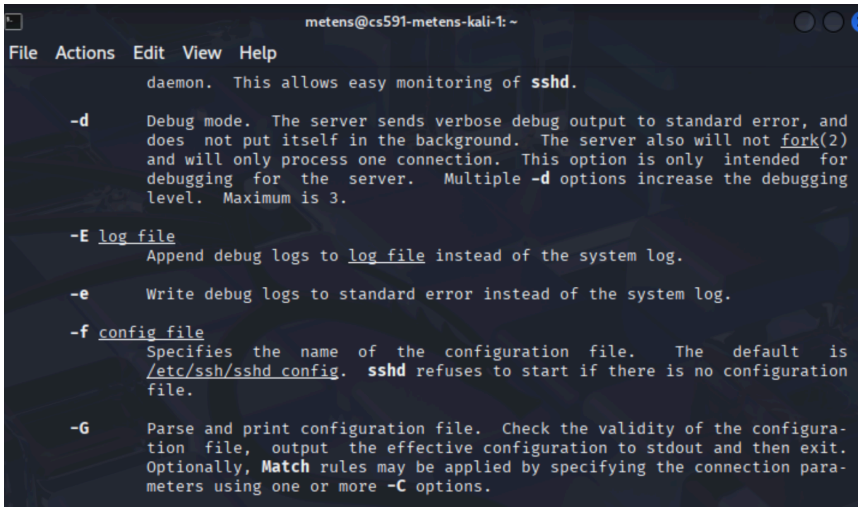
This part brought me a bunch of issues. I was having trouble with alias and my backdoor ls just wasn't working no matter what I tried:

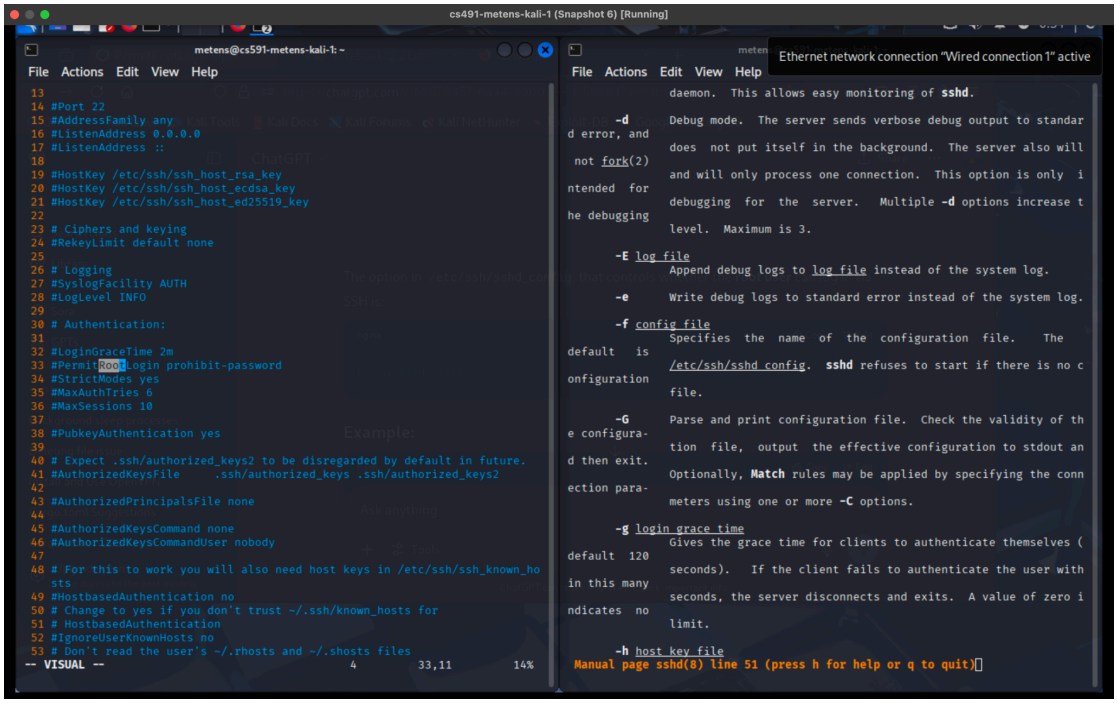


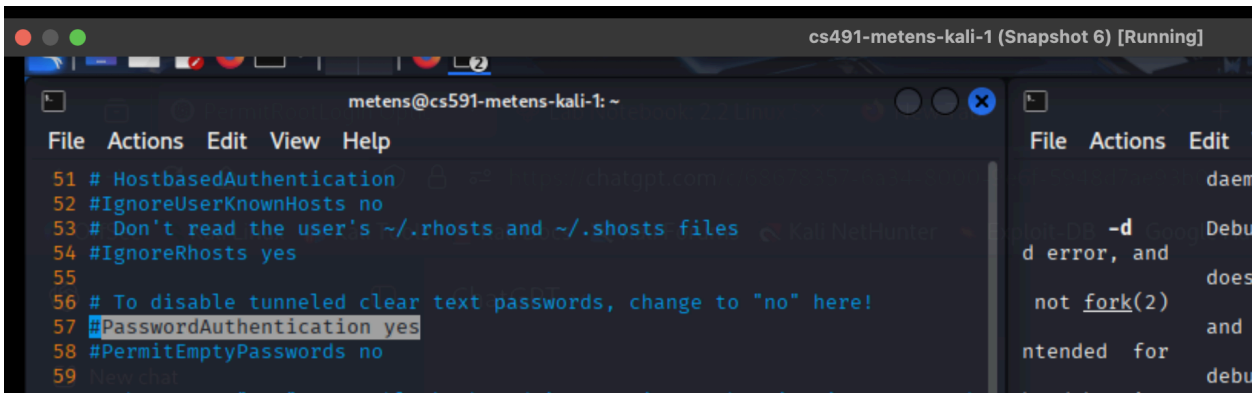
Here are the steps I followed to resolve the issue. I just rechecked my steps, and somehow it worked after that:



Task 10: sshd


metens@cs591-metens-kali-1: ~
File Actions Edit View Help
daemon. This allows easy monitoring of sshd.
-d Debug mode. The server sends verbose debug output to standard error, and does not put itself in the background. The server also will not fork(2) and will only process one connection. This option is only intended for debugging for the server. Multiple -d options increase the debugging level. Maximum is 3.
-E log file
Append debug logs to log file instead of the system log.
-e Write debug logs to standard error instead of the system log.
-f config file
Specifies the name of the configuration file. The default is /etc/ssh/sshd_config. sshd refuses to start if there is no configuration file.
-G Parse and print configuration file. Check the validity of the configuration file, output the effective configuration to stdout and then exit. Optionally, Match rules may be applied by specifying the connection parameters using one or more -C options.

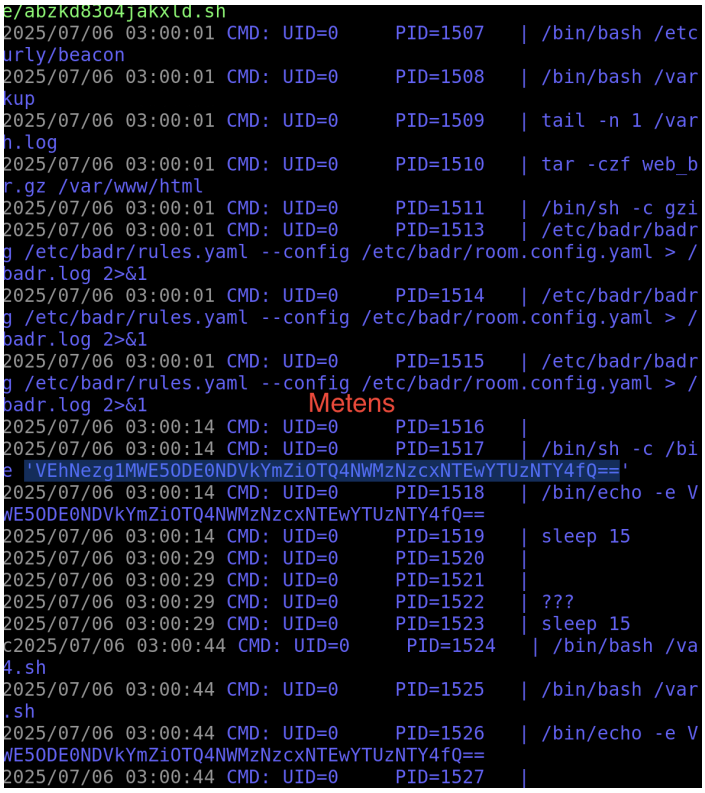

cs491-metens-kali-1 (Snapshot 6) [Running]
metens@cs591-metens-kali-1: ~
File Actions Edit View Help
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
Ciphers and keying
#RekeyLimit default none
Logging
#SyslogFacility AUTH
#LogLevel INFO
Authentication:
#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
#PubkeyAuthentication yes
Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2
#AuthorizedPrincipalsFile none
#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody
For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
Change to yes if you don't trust ~/.ssh/known_hosts for
HostbasedAuthentication
#IgnoreUserKnownHosts no
Don't read the user's ~/.rhosts and ~/.shosts files
-- VISUAL --
Ethernet network connection "Wired connection 1" active
daemon. This allows easy monitoring of sshd.
-d Debug mode. The server sends verbose debug output to standard error, and does not put itself in the background. The server also will not fork(2) and will only process one connection. This option is only intended for debugging for the server. Multiple -d options increase the debugging level. Maximum is 3.
-E log file
Append debug logs to log file instead of the system log.
-e Write debug logs to standard error instead of the system log.
-f config file
Specifies the name of the configuration file. The default is /etc/ssh/sshd_config. sshd refuses to start if there is no configuration file.
-G Parse and print configuration file. Check the validity of the configuration file, output the effective configuration to stdout and then exit. Optionally, Match rules may be applied by specifying the connection parameters using one or more -C options.
-g login_grace_time
Gives the grace time for clients to authenticate themselves (default 120 seconds). If the client fails to authenticate the user within this many seconds, the server disconnects and exits. A value of zero indicates no limit.
-h host_key_file
Manual page sshd(8) line 51 (press h for help or q to quit)


cs491-metens-kali-1 (Snapshot 6) [Running]
metens@cs591-metens-kali-1: ~
File Actions Edit View Help
51 # HostbasedAuthentication
52 #IgnoreUserKnownHosts no
53 # Don't read the user's ~/.rhosts and ~/.shosts files
54 #IgnoreRhosts yes
55
56 # To disable tunneled clear text passwords, change to "no" here!
57 #PasswordAuthentication yes
58 #PermitEmptyPasswords no
59
File Actions Edit
-d
d error, and
not fork(2)
ntended for

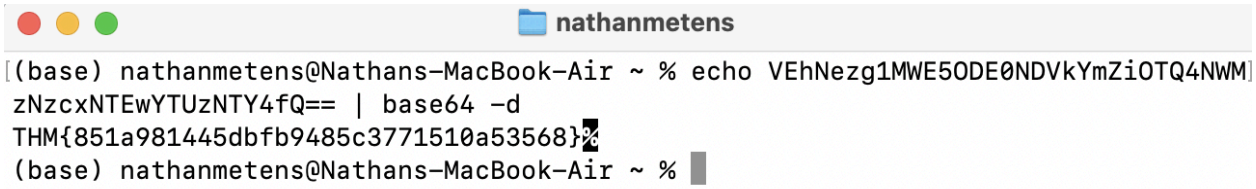
The name of the option that allows one to log in as root is on line 33 of the sshd_config file. It is called “PermitRootLogin”. The value can be changed to “yes”. Similarly, the name of the option that permits authentication via password is on line 57 and is called “PasswordAuthentication”; it is set to “yes”.

Task 11: TryHackMe Linux process analysis

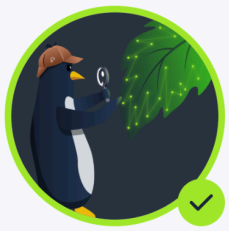
One of the cool things I did in this task was to list the `pspy64` command, and every 15 seconds, the system outputed this:



This was an encoded message. To decode the flag, I ran this in my terminal:



The rest of this TryHackMe room took me three days to complete. I got stuck on a task for so long, and since I’m such a slow reader, it took me a total of about 7 hours to complete.



Congratulations on completing Linux Process Analysis!!! 🎉

Points earned	Completed tasks	Room type	Difficulty	Streak
96	8	Walkthrough	Easy	3

This room counted toward joining the league 🎯

Metens

July 2, 2025 at 2:48 PM

Metens

metens [0x2][APPRENTICE]

Get profile badge ID | Share room badges

Rank	Badges	Streak	Completed rooms
1006473	1	3	5

Completed rooms | Certificates | Skills matrix | Badges | Created rooms | Yearly activity | Tickets

Offensive Security Intro
Hack your first website (legally in a safe environment) and experience an ethical hacker's job.
Easy
Free | Walkthrough

OpenVPN
A guide to connecting to our network using OpenVPN.
Easy
Free | Walkthrough

Linux Fundamentals Part 1
Embark on the journey of learning the fundamentals of Linux. Learn to run some of the first essential commands on an interactive terminal.
Info
Free | Walkthrough

Linux File System Analysis
Perform real-time file system analysis on a Linux system to identify an attacker's artefacts.
Easy
Free | Walkthrough

Linux Process Analysis
Perform thorough process and application analysis to identify an attacker's persistence methods.
Easy
Free | Walkthrough