Fraser Rankin
101192297

COMP 4108 Assignment 2

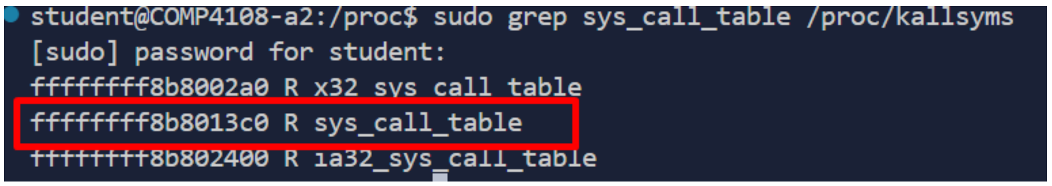
Due 11:59PM on Oct 8

Assignment out of 47 marks total

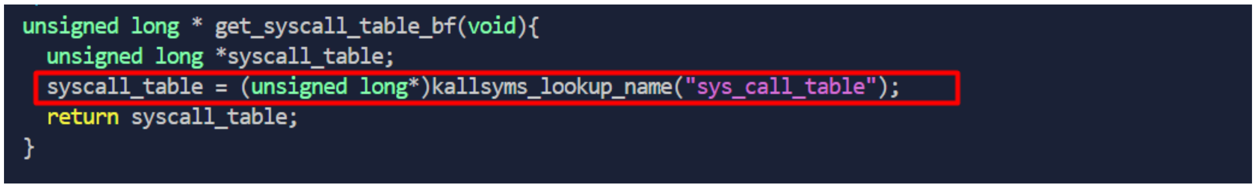
Files in Submission:

1. fraserrankin-assignment2.pdf
2. fraserrankin-assignment2.zip
    a. rootkit.c
    b. Makefile
    c. eject.sh
    d. insert.sh
    e. insert B.txt (a text file containing the code I used for part B)
    f. new_getdents64().txt (text file containing the code I used for C1)

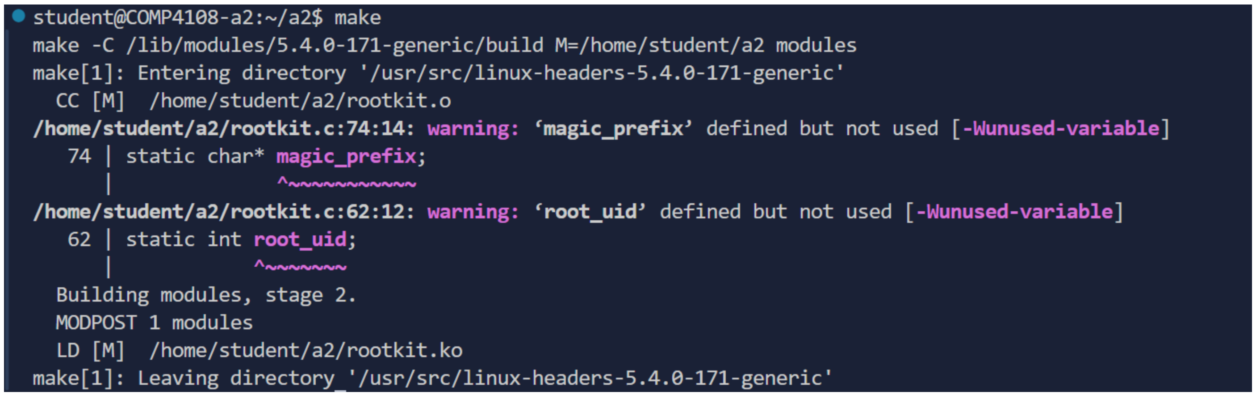
Part A - Setup (7 marks)

1. wget --user comp4108 --password z48QVUanF2wYV49A https://www.cisl.carleton.ca/~hpatel/comp4108/private/code/a2/a2.tar.gz
   tar -xvzf a2.tar.gz
2. Command: sudo bash
   and then enter the password
3. 

```
student@COMP4108-a2:/proc$ sudo grep sys_call_table /proc/kallsyms
[sudo] password for student:
ffffffff8b8002a0 R x32_sys_call_table
ffffffff8b8013c0 R sys_call_table
ffffffff8b802400 R ia32_sys_call_table
```

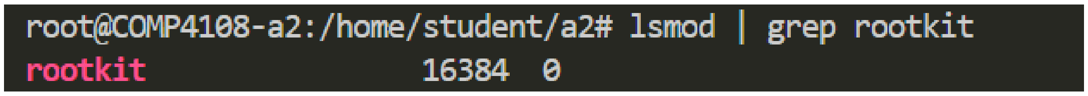
4. 

```
unsigned long * get_syscall_table_bf(void){
  unsigned long *syscall_table;
  syscall_table = (unsigned long*)kallsyms_lookup_name("sys_call_table");
  return syscall_table;
}
```

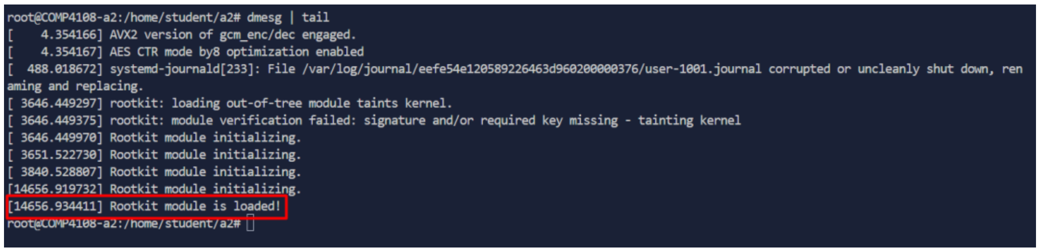
5. 

```
student@COMP4108-a2:~/a2$ make
make -C /lib/modules/5.4.0-171-generic/build M=/home/student/a2 modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-171-generic'
  CC [M]  /home/student/a2/rootkit.o
/home/student/a2/rootkit.c:74:14: warning: ‘magic_prefix’ defined but not used [-Wunused-variable]
   74 | static char* magic_prefix;
      |              ^~~~~~~~~~~~
/home/student/a2/rootkit.c:62:12: warning: ‘root_uid’ defined but not used [-Wunused-variable]
   62 | static int root_uid;
      |            ^~~~~~~~
  Building modules, stage 2.
  MODPOST 1 modules
  LD [M]  /home/student/a2/rootkit.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-171-generic'
```

6. I had added the protect_memory() and unprotect_memory(), where it was commented in the comments of the rootkit.c file



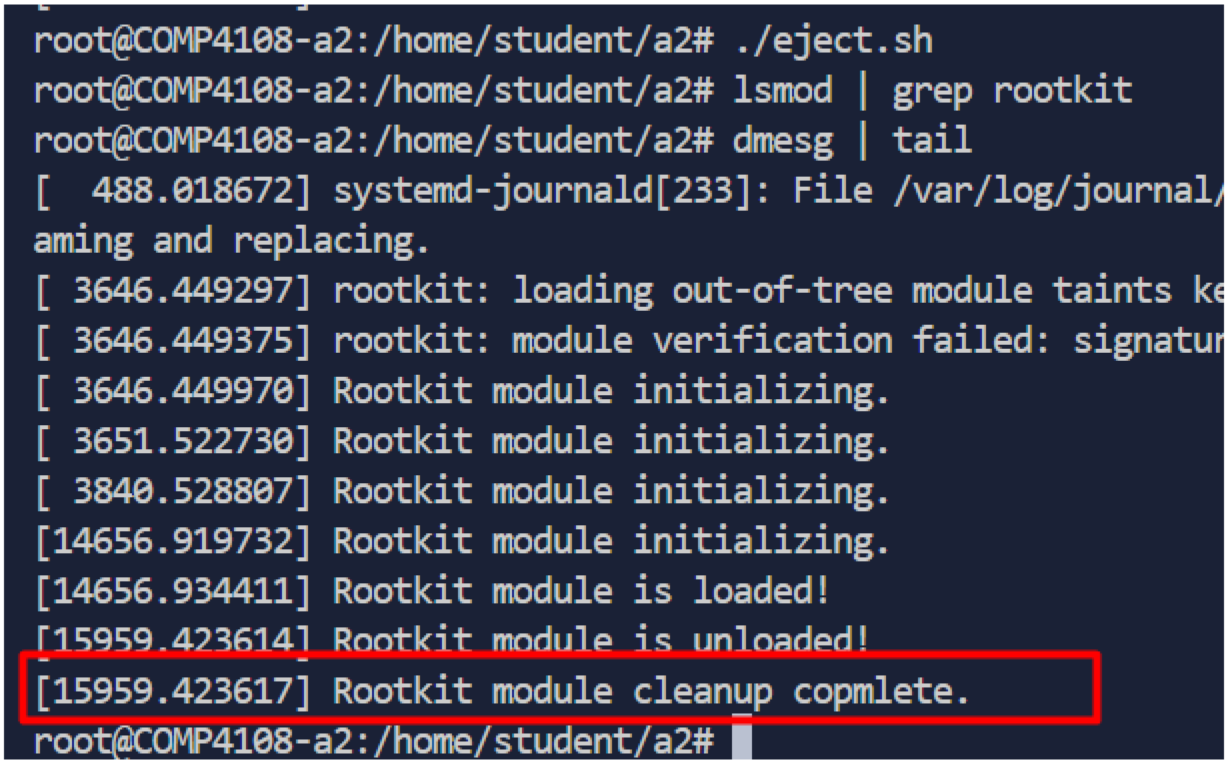
```
root@COMP4108-a2:/home/student/a2# lsmod | grep rootkit
rootkit                16384  0
```



```
root@COMP4108-a2:/home/student/a2# dmesg | tail
[    4.354166] AVX2 version of gcm_enc/dec engaged.
[    4.354167] AES CTR mode by8 optimization enabled
[  488.018672] systemd-journald[233]: File /var/log/journal/eefe54e120589226463d960200000376/user-1001.journal corrupted or uncleanly shut down, ren
aming and replacing.
[ 3646.449297] rootkit: loading out-of-tree module taints kernel.
[ 3646.449375] rootkit: module verification failed: signature and/or required key missing - tainting kernel
[ 3646.449970] Rootkit module initializing.
[ 3651.522730] Rootkit module initializing.
[ 3840.528807] Rootkit module initializing.
[14656.919732] Rootkit module initializing.
[14656.934411] Rootkit module is loaded!
root@COMP4108-a2:/home/student/a2# 
```

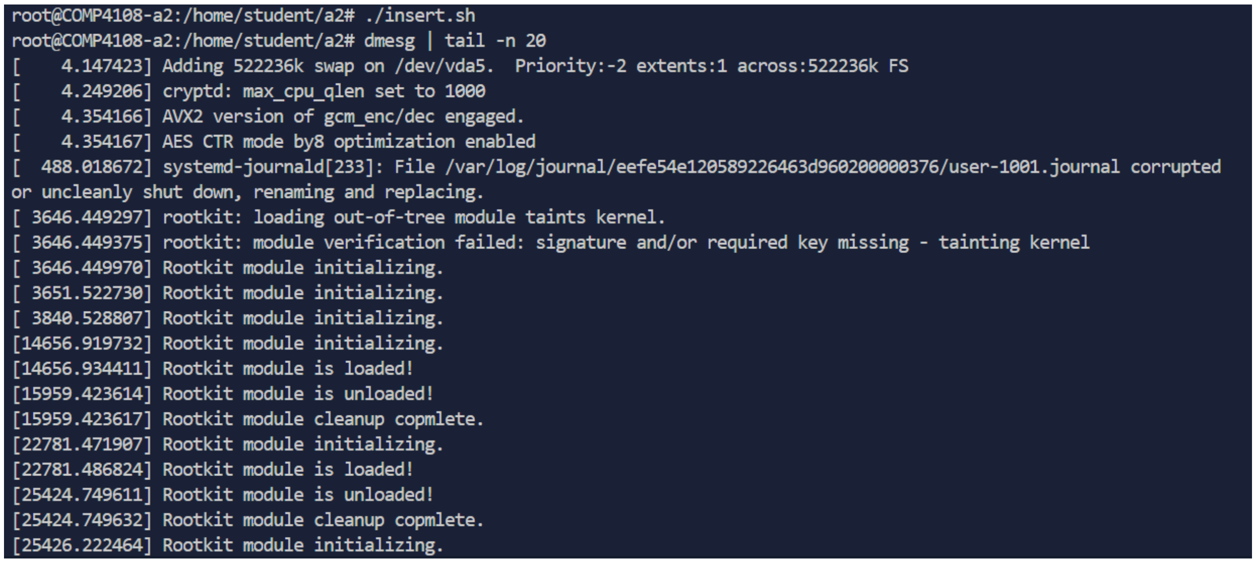
7.

I uncommented the lines of code where they were labeled in the comments of the rookit.c file

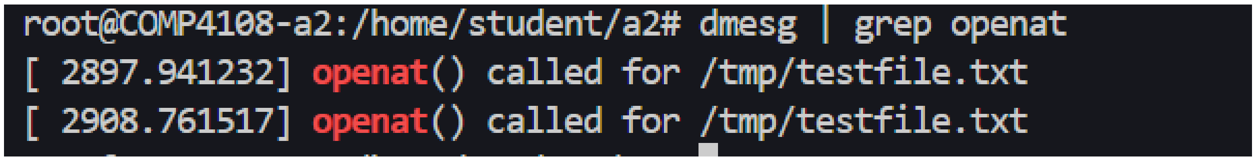


```
root@COMP4108-a2:/home/student/a2# ./eject.sh
root@COMP4108-a2:/home/student/a2# lsmod | grep rootkit
root@COMP4108-a2:/home/student/a2# dmesg | tail
[  488.018672] systemd-journald[233]: File /var/log/journal/
aming and replacing.
[ 3646.449297] rootkit: loading out-of-tree module taints ke
[ 3646.449375] rootkit: module verification failed: signatur
[ 3646.449970] Rootkit module initializing.
[ 3651.522730] Rootkit module initializing.
[ 3840.528807] Rootkit module initializing.
[14656.919732] Rootkit module initializing.
[14656.934411] Rootkit module is loaded!
[15959.423614] Rootkit module is unloaded!
[15959.423617] Rootkit module cleanup copmlete.
root@COMP4108-a2:/home/student/a2# 
```

8.



```
root@COMP4108-a2:/home/student/a2# ./insert.sh
root@COMP4108-a2:/home/student/a2# dmesg | tail -n 20
[    4.147423] Adding 522236k swap on /dev/vda5.  Priority:-2 extents:1 across:522236k FS
[    4.249206] cryptd: max_cpu_qlen set to 1000
[    4.354166] AVX2 version of gcm_enc/dec engaged.
[    4.354167] AES CTR mode by8 optimization enabled
[  488.018672] systemd-journald[233]: File /var/log/journal/eefe54e120589226463d960200000376/user-1001.journal corrupted
or uncleanly shut down, renaming and replacing.
[ 3646.449297] rootkit: loading out-of-tree module taints kernel.
[ 3646.449375] rootkit: module verification failed: signature and/or required key missing - tainting kernel
[ 3646.449970] Rootkit module initializing.
[ 3651.522730] Rootkit module initializing.
[ 3840.528807] Rootkit module initializing.
[14656.919732] Rootkit module initializing.
[14656.934411] Rootkit module is loaded!
[15959.423614] Rootkit module is unloaded!
[15959.423617] Rootkit module cleanup copmlete.
[22781.471907] Rootkit module initializing.
[22781.486824] Rootkit module is loaded!
[25424.749611] Rootkit module is unloaded!
[25424.749632] Rootkit module cleanup copmlete.
[25426.222464] Rootkit module initializing.
```



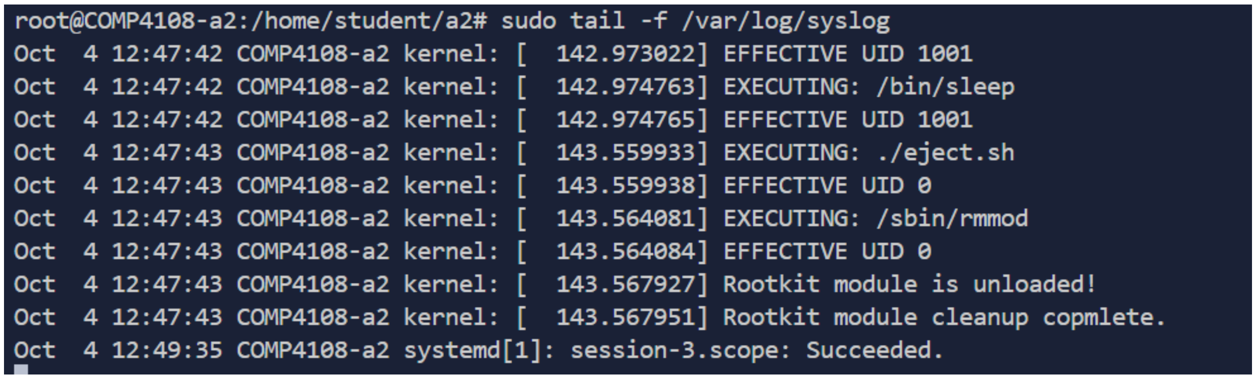
```
root@COMP4108-a2:/home/student/a2# dmesg | grep openat
[ 2897.941232] openat() called for /tmp/testfile.txt
[ 2908.761517] openat() called for /tmp/testfile.txt
```

9.

Rootkits are designed to take advantage of the systems and gain the entity that deployed the rootkit, accees to the system that the rootkit was deployed on. However, there are several security principles that can be implemented to help prevent the use of rootkits on our systems. The first principle that can be implemented is P4 Complete Mediation. The principle focuses on the verification of entities before anything is run on the system, as well as verifying the integrity of files. This would help us with the rootkit as it would force the verification of both the user that implemented the rootkit but also verify the contents of files that are being changed by the rootkit. Another principle that would help block the rootkits is P5 Isolated-Compartments. This principle follows the logic of isolating different functions and preventing cross-system changes by a singular program. This would stop the rootkit from making changes to any important files.

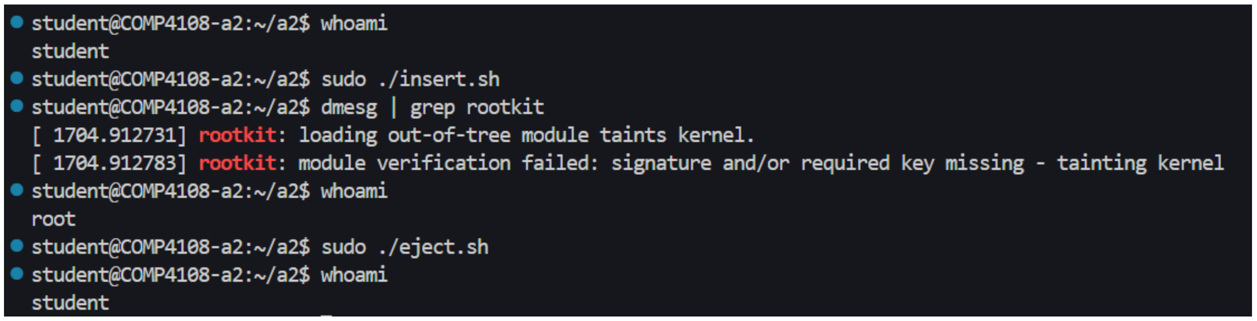
Part B - Backdoor (15 Marks)

1. To make a new hook for the execve syscall, I followed a path like how the openat syscall was made in the rootkit. I declared the filename and the return value; I then allocated the filename in kernel memory and copied the filename into the kernel variable. I then declared the effective user id as current_euid() to get the euid of the current process. I then printed the info to the system logs with printk and then freed the filename from kernel memory, set the return value as original_execve(regs) and returned the returned value.

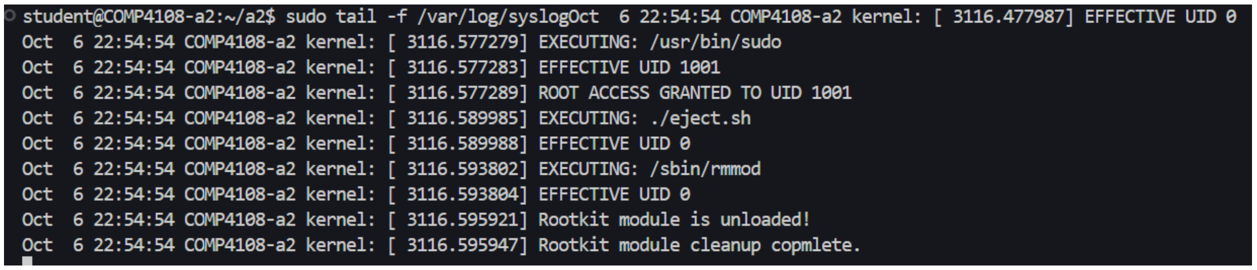


```
root@COMP4108-a2:/home/student/a2# sudo tail -f /var/log/syslog
Oct  4 12:47:42 COMP4108-a2 kernel: [  142.973022] EFFECTIVE UID 1001
Oct  4 12:47:42 COMP4108-a2 kernel: [  142.974763] EXECUTING: /bin/sleep
Oct  4 12:47:42 COMP4108-a2 kernel: [  142.974765] EFFECTIVE UID 1001
Oct  4 12:47:43 COMP4108-a2 kernel: [  143.559933] EXECUTING: ./eject.sh
Oct  4 12:47:43 COMP4108-a2 kernel: [  143.559938] EFFECTIVE UID 0
Oct  4 12:47:43 COMP4108-a2 kernel: [  143.564081] EXECUTING: /sbin/rmmod
Oct  4 12:47:43 COMP4108-a2 kernel: [  143.564084] EFFECTIVE UID 0
Oct  4 12:47:43 COMP4108-a2 kernel: [  143.567927] Rootkit module is unloaded!
Oct  4 12:47:43 COMP4108-a2 kernel: [  143.567951] Rootkit module cleanup copmlete.
Oct  4 12:49:35 COMP4108-a2 systemd[1]: session-3.scope: Succeeded.
```

2.



```
student@COMP4108-a2:~/a2$ whoami
student
student@COMP4108-a2:~/a2$ sudo ./insert.sh
student@COMP4108-a2:~/a2$ dmesg | grep rootkit
[ 1704.912731] rootkit: loading out-of-tree module taints kernel.
[ 1704.912783] rootkit: module verification failed: signature and/or required key missing - tainting kernel
student@COMP4108-a2:~/a2$ whoami
root
student@COMP4108-a2:~/a2$ sudo ./eject.sh
student@COMP4108-a2:~/a2$ whoami
student
```

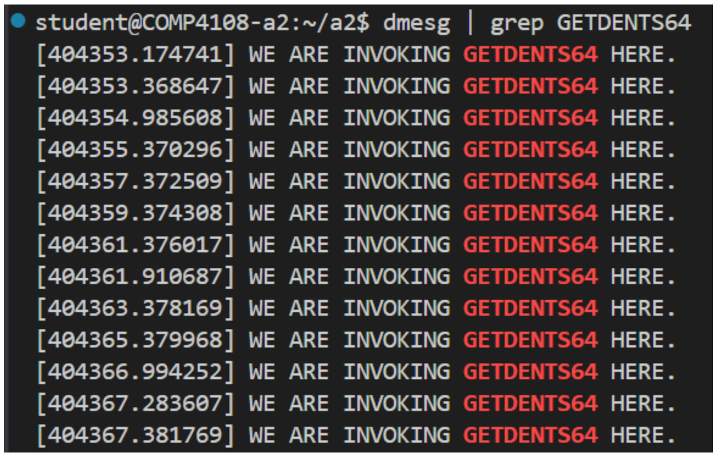


```
student@COMP4108-a2:~/a2$ sudo tail -f /var/log/syslogOct  6 22:54:54 COMP4108-a2 kernel: [ 3116.477987] EFFECTIVE UID 0
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.577279] EXECUTING: /usr/bin/sudo
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.577283] EFFECTIVE UID 1001
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.577289] ROOT ACCESS GRANTED TO UID 1001
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.589985] EXECUTING: ./eject.sh
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.589988] EFFECTIVE UID 0
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.593802] EXECUTING: /sbin/rmmod
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.593804] EFFECTIVE UID 0
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.595921] Rootkit module is unloaded!
Oct  6 22:54:54 COMP4108-a2 kernel: [ 3116.595947] Rootkit module cleanup copmlete.
```

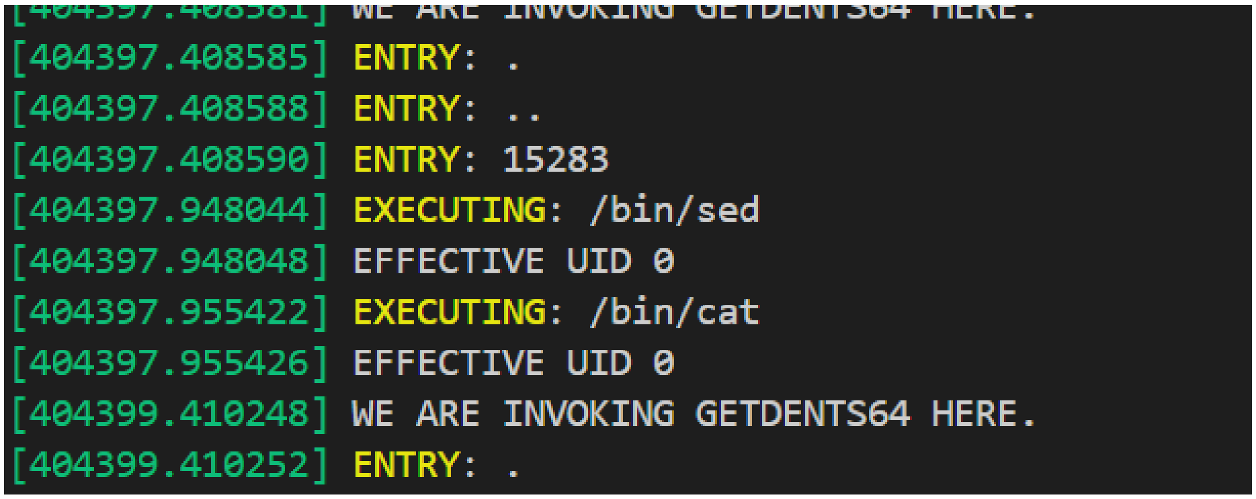
From B1 to B2 we made some changes to our rootkit.c code, specifically to the the new new_execve() function that we hade created previously to print the name of the files being executed and the EUID of the user running the file. The function works by replacing the systems execve system calls (we intercept and get the file name and EUID of the process executing the command). We first check if the EUID of the process is root ID, we then escalate the privileges of the processes' privileges to root. Next, the prepare_kernel_cred() is used to create new credentials and the EUID, UID, GID, and EGID to root and then use the commit_creds() function to commit them. We print the changes using printk to the kernel log. The allocated memory is freed via kfree() and we return the original execve syscall. (add insert.sh changes here)

Part C

1. First, I made a return variable to hold the original getdents64 syscall, along with a linux_dirent64 struct, an unsigned long for an offset and string (list of chars) as the buffer to keep track of the directory entries. After we set our return value to the original getdents, we make sure that the function does not return 0 via error handling. Afterwards, we allocate memory in kernel space for all the directory entries in the original getdents function, then we copy the directory entries from user space using the copy_form_user() function to copy the data into the buffer we have made. Then we log the message that we are invoking the getdents function with printk, so it is displayed in the kernel log. Afterwards, we iterate through the directory entries, and we use the linux_dirent64 struct we made earlier to be able to access the information from the directory entry, such as its name. We then print that info back to the kernel log via printk. We use the offset variable we made earlier to help us iterate through the entries. Finally, we free all the memory that we have allocated and return the original getdents function. The hook function will intercept the original getdents function when it is called and use our new_getdents function and then afterwards resume the normal process of calling the getdents function. (see the txt file associated with this question to see the what it looked like before we changed it for Q23)

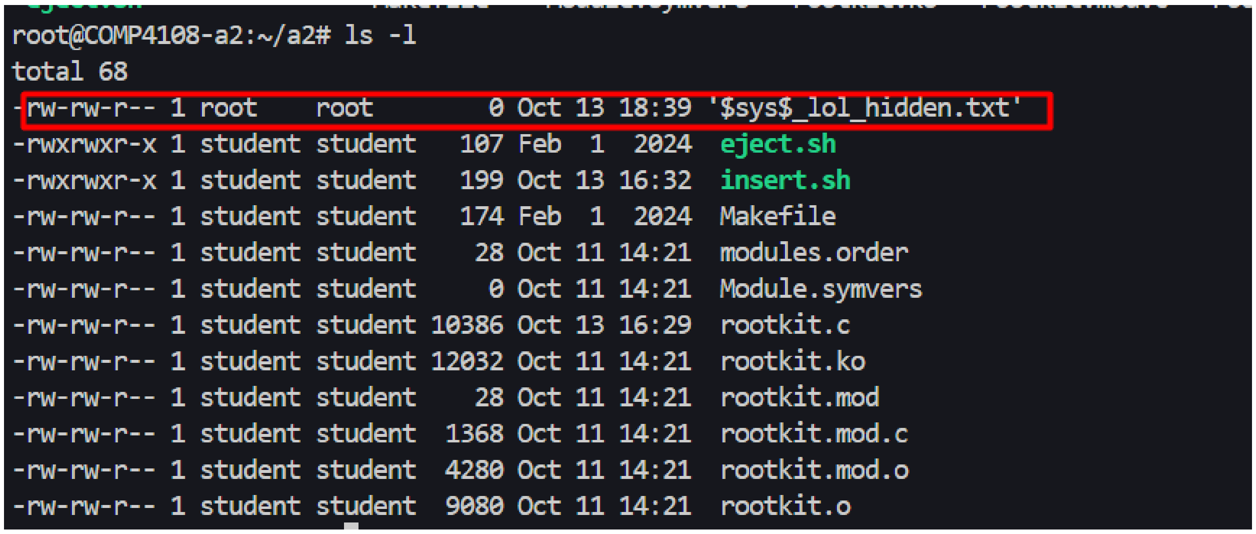


```
student@COMP4108-a2:~/a2$ dmesg | grep GETDENTS64
[404353.174741] WE ARE INVOKING GETDENTS64 HERE.
[404353.368647] WE ARE INVOKING GETDENTS64 HERE.
[404354.985608] WE ARE INVOKING GETDENTS64 HERE.
[404355.370296] WE ARE INVOKING GETDENTS64 HERE.
[404357.372509] WE ARE INVOKING GETDENTS64 HERE.
[404359.374308] WE ARE INVOKING GETDENTS64 HERE.
[404361.376017] WE ARE INVOKING GETDENTS64 HERE.
[404361.910687] WE ARE INVOKING GETDENTS64 HERE.
[404363.378169] WE ARE INVOKING GETDENTS64 HERE.
[404365.379968] WE ARE INVOKING GETDENTS64 HERE.
[404366.994252] WE ARE INVOKING GETDENTS64 HERE.
[404367.283607] WE ARE INVOKING GETDENTS64 HERE.
[404367.381769] WE ARE INVOKING GETDENTS64 HERE.
```

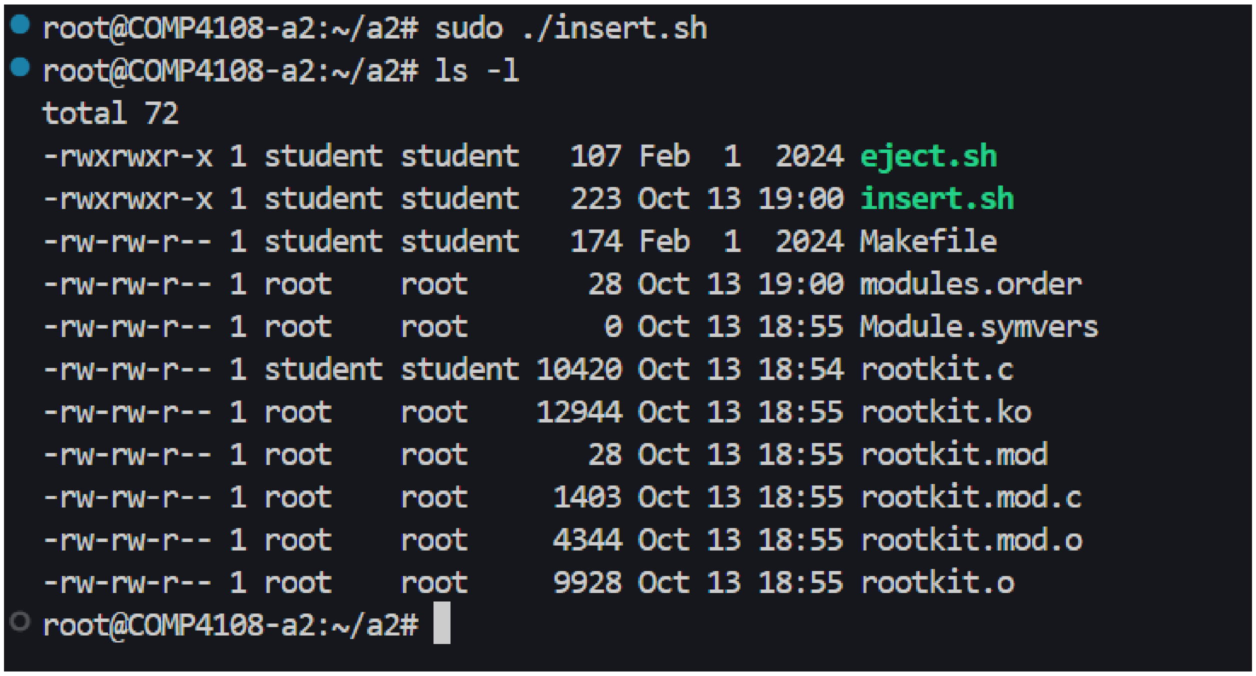


```
[404397.408581] WE ARE INVOKING GETDENTS64 HERE.
[404397.408585] ENTRY: .
[404397.408588] ENTRY: ..
[404397.408590] ENTRY: 15283
[404397.948044] EXECUTING: /bin/sed
[404397.948048] EFFECTIVE UID 0
[404397.955422] EXECUTING: /bin/cat
[404397.955426] EFFECTIVE UID 0
[404399.410248] WE ARE INVOKING GETDENTS64 HERE.
[404399.410252] ENTRY: .
```

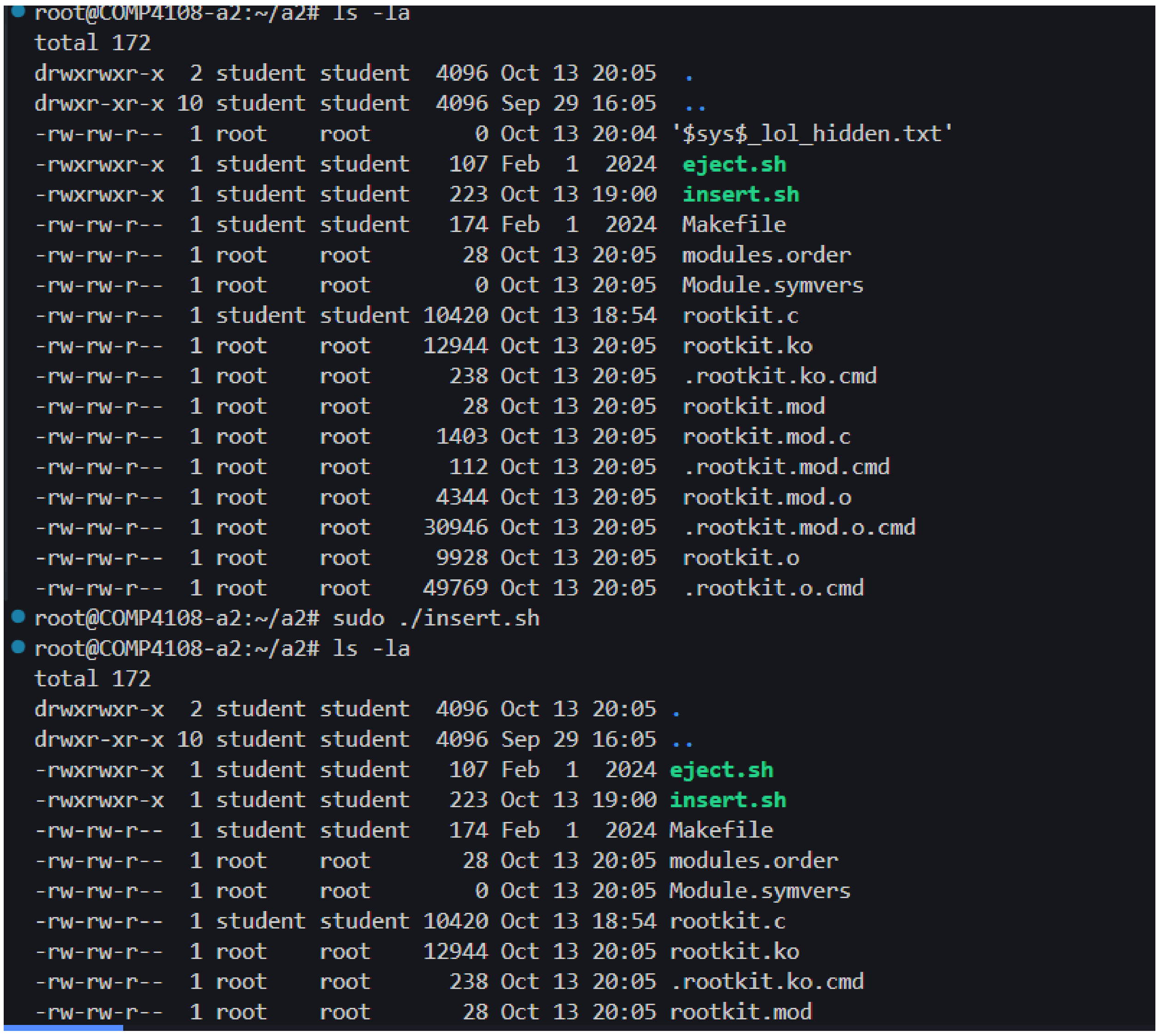
2. Like with for the previous question, this part starts with intercepting the original getdents() function with our own. In this question, we also finally defined the magic_prefix function that we have been ignoring up to this point. This lets us grab the value that is used in our insert.sh file, dubbed our “special prefix”. Next, we hook the original getdents function with our new_getdents function so that we can make our necessary changes. In the new getdents function, we first make a bunch of variables, including three linux_dirent structs, one to keep track of the user space directory, and the other two will be used to help us traverse the and make changes to the list of directory entries. Then, like the previous version we made, we grab the original value of the original getdents function, make sure it is not empty, and allocate memory in kernel space for when we need to iterate through the directory entries. Again, like the previous iteration, we use the copy_from_user() function to get the directory entries into kernel buffer. We then iterate through the directory entries and check if files with the suffix exist, if they do, then we remove that entry. After, we then free the memory we have allocated and then we return the original getdents function. In the insert.sh function, we have edited it to now grab the magic prefix.



```
root@COMP4108-a2:~/a2# ls -l
total 68
-rw-rw-r-- 1 root    root        0 Oct 13 18:39 '$sys$_lol_hidden.txt'
-rwxrwxr-x 1 student student   107 Feb  1  2024  eject.sh
-rwxrwxr-x 1 student student   199 Oct 13 16:32  insert.sh
-rw-rw-r-- 1 student student   174 Feb  1  2024  Makefile
-rw-rw-r-- 1 student student    28 Oct 11 14:21  modules.order
-rw-rw-r-- 1 student student     0 Oct 11 14:21  Module.symvers
-rw-rw-r-- 1 student student 10386 Oct 13 16:29  rootkit.c
-rw-rw-r-- 1 student student 12032 Oct 11 14:21  rootkit.ko
-rw-rw-r-- 1 student student    28 Oct 11 14:21  rootkit.mod
-rw-rw-r-- 1 student student  1368 Oct 11 14:21  rootkit.mod.c
-rw-rw-r-- 1 student student  4280 Oct 11 14:21  rootkit.mod.o
-rw-rw-r-- 1 student student  9080 Oct 11 14:21  rootkit.o
```



```
root@COMP4108-a2:~/a2# sudo ./insert.sh
root@COMP4108-a2:~/a2# ls -l
total 72
-rwxrwxr-x 1 student student   107 Feb  1  2024 eject.sh
-rwxrwxr-x 1 student student   223 Oct 13 19:00 insert.sh
-rw-rw-r-- 1 student student   174 Feb  1  2024 Makefile
-rw-rw-r-- 1 root    root       28 Oct 13 19:00 modules.order
-rw-rw-r-- 1 root    root        0 Oct 13 18:55 Module.symvers
-rw-rw-r-- 1 student student 10420 Oct 13 18:54 rootkit.c
-rw-rw-r-- 1 root    root    12944 Oct 13 18:55 rootkit.ko
-rw-rw-r-- 1 root    root       28 Oct 13 18:55 rootkit.mod
-rw-rw-r-- 1 root    root     1403 Oct 13 18:55 rootkit.mod.c
-rw-rw-r-- 1 root    root     4344 Oct 13 18:55 rootkit.mod.o
-rw-rw-r-- 1 root    root     9928 Oct 13 18:55 rootkit.o
root@COMP4108-a2:~/a2# 
```



```
root@COMP4108-a2:~/a2# ls -la
total 172
drwxrwxr-x  2 student student  4096 Oct 13 20:05  .
drwxr-xr-x 10 student student  4096 Sep 29 16:05  ..
-rw-rw-r--  1 root    root        0 Oct 13 20:04 '$sys$_lol_hidden.txt'
-rwxrwxr-x  1 student student   107 Feb  1  2024  eject.sh
-rwxrwxr-x  1 student student   223 Oct 13 19:00  insert.sh
-rw-rw-r--  1 student student   174 Feb  1  2024  Makefile
-rw-rw-r--  1 root    root       28 Oct 13 20:05  modules.order
-rw-rw-r--  1 root    root        0 Oct 13 20:05  Module.symvers
-rw-rw-r--  1 student student 10420 Oct 13 18:54  rootkit.c
-rw-rw-r--  1 root    root    12944 Oct 13 20:05  rootkit.ko
-rw-rw-r--  1 root    root      238 Oct 13 20:05  .rootkit.ko.cmd
-rw-rw-r--  1 root    root       28 Oct 13 20:05  rootkit.mod
-rw-rw-r--  1 root    root     1403 Oct 13 20:05  rootkit.mod.c
-rw-rw-r--  1 root    root      112 Oct 13 20:05  .rootkit.mod.cmd
-rw-rw-r--  1 root    root     4344 Oct 13 20:05  rootkit.mod.o
-rw-rw-r--  1 root    root    30946 Oct 13 20:05  .rootkit.mod.o.cmd
-rw-rw-r--  1 root    root     9928 Oct 13 20:05  rootkit.o
-rw-rw-r--  1 root    root    49769 Oct 13 20:05  .rootkit.o.cmd
root@COMP4108-a2:~/a2# sudo ./insert.sh
root@COMP4108-a2:~/a2# ls -la
total 172
drwxrwxr-x  2 student student  4096 Oct 13 20:05 .
drwxr-xr-x 10 student student  4096 Sep 29 16:05 ..
-rwxrwxr-x  1 student student   107 Feb  1  2024 eject.sh
-rwxrwxr-x  1 student student   223 Oct 13 19:00 insert.sh
-rw-rw-r--  1 student student   174 Feb  1  2024 Makefile
-rw-rw-r--  1 root    root       28 Oct 13 20:05 modules.order
-rw-rw-r--  1 root    root        0 Oct 13 20:05 Module.symvers
-rw-rw-r--  1 student student 10420 Oct 13 18:54 rootkit.c
-rw-rw-r--  1 root    root    12944 Oct 13 20:05 rootkit.ko
-rw-rw-r--  1 root    root      238 Oct 13 20:05 .rootkit.ko.cmd
-rw-rw-r--  1 root    root       28 Oct 13 20:05 rootkit.mod
```