# **Project**

## **Team Members**

- Mahmoud Mostafa Zaki
- Randa Mohamed Hassan

## Adding A System Call To The Linux Kernel (5.8.1) In Ubuntu (20.04 LTS)

1. Fully update operating system.

#### sudo apt update && sudo apt upgrade -y

2. Download and install the essential packages to compile kernels.

sudo apt install build-essential libncurses-dev libssl-dev libelf-dev bison flex -y

#### sudo apt install vim -y

3. Clean up installed packages.

#### sudo apt clean && sudo apt autoremove -y

4. Download the source code of the version of the Linux kernel (*which is* 5.8.1) to home folder.

#### wget -P ~/ https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.8.1.tar.xz

5. Unpack the tarball just downloaded to home folder.

#### tar -xvf ~/linux-5.8.1.tar.xz -C ~/

6. Check the version of current kernel.

#### uname -r

display 5.4.0-42-generic

7. Change working directory to the root directory of the recently unpacked source code.

#### cd ~/linux-5.8.1/

8. Create the home directory of system call.

Decide a name for system call, and keep it consistent from this point onwards. I have chosen identity.

#### mkdir identity

9. Create a C file for system call.

Create the C file with the following command.

#### nano identity/identity.c

Write the following code in it.

```
#include #include kernel.h>
#include kernel.h>

SYSCALL_DEFINE0(identity)

{
    printk("we are Mahmoud and randa.\n");
    return 0;
}
```

10. Create a Makefile for system call.

Create the Makefile with the following command.

#### nano identity/Makefile

Write the following code in it.

#### obj-y := identity.o

11. Add the home directory of system call to the main Makefile of the kernel.

Open the Makefile with the following command.

#### nano Makefile

Search for core-y. In the second result, The series of directories.

kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/

Add the home directory of system call at the end like the following.

#### kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/ identity/

12. Add a corresponding function prototype for system call to the header file of system calls.

Open the header file with the following command.

#### nano include/linux/syscalls.h

Navigate to the bottom of it and write the following code just above #endif

#### asmlinkage long sys\_identity(void);

13. Add system call to the kernel's system call table.

#### nano arch/x86/entry/syscalls/syscall\_64.tbl

Navigate to the bottom of it. find a series of x32 system calls. Scroll to the section above it. This is the section of interest. Add the following code at the end of this section respecting the chronology of the row as well as the format of the column. Use Tab for space.

#### 440 common identity sys\_identity

14. Configure the kernel.

The window of terminal must maximized.

Open the configuration window with the following command.

#### make menuconfig

Use **Tab** to move between options. Make no changes to keep it in default settings.

15. Find out how many logical cores.

#### nproc

The following few commands require a long time to be executed. Parallel processing will greatly speed them up. For me, it is 2. Therefore, I will put 2 after -j in the following commands.

16. write these commands

#### make clean

make mrproper

make menuconfig

make localmodconfig

choose yes then no then yes

17. Compile the kernel's source code.

#### make -j2

18. Prepare the installer of the kernel.

#### sudo make modules\_install -j2

19. Install the kernel.

#### sudo make install -j2

20. Update the bootloader of the operating system with the new kernel.

#### sudo update-grub

21. Restart

#### shutdown –r now

22. Check the version of your current kernel.

#### uname -r

Display 5.8.1

23. Change working directory to home directory.

#### cd ~

24. Create a C file to generate a report of the success or failure of system call.

#### nano report.c

Write the following code in it.

```
#include ux/kernel.h>
#include <sys/syscall.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>
#define __NR_identity 440
long identity_syscall(void)
  return syscall(__NR_identity);
int main(int argc, char *argv[])
  long activity;
  activity = identity_syscall();
  if(activity < 0)
     perror("Sorry, mahmoud and randa.");
  else
     printf("Congratulations, mahmoud and randa! \n");
  return 0;
```

25. Compile the C file just created.

gcc -o report report.c

26. Run the C file just compiled.

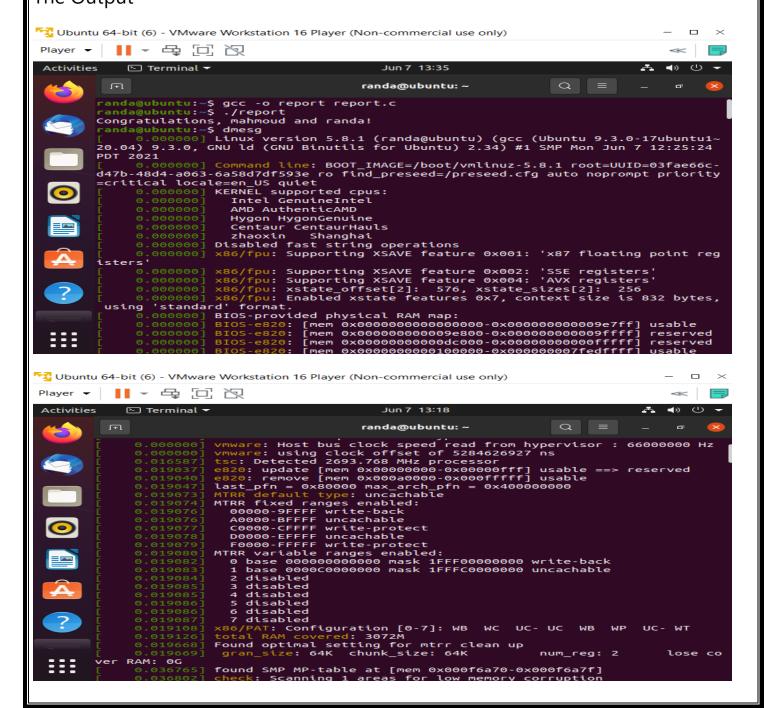
./report

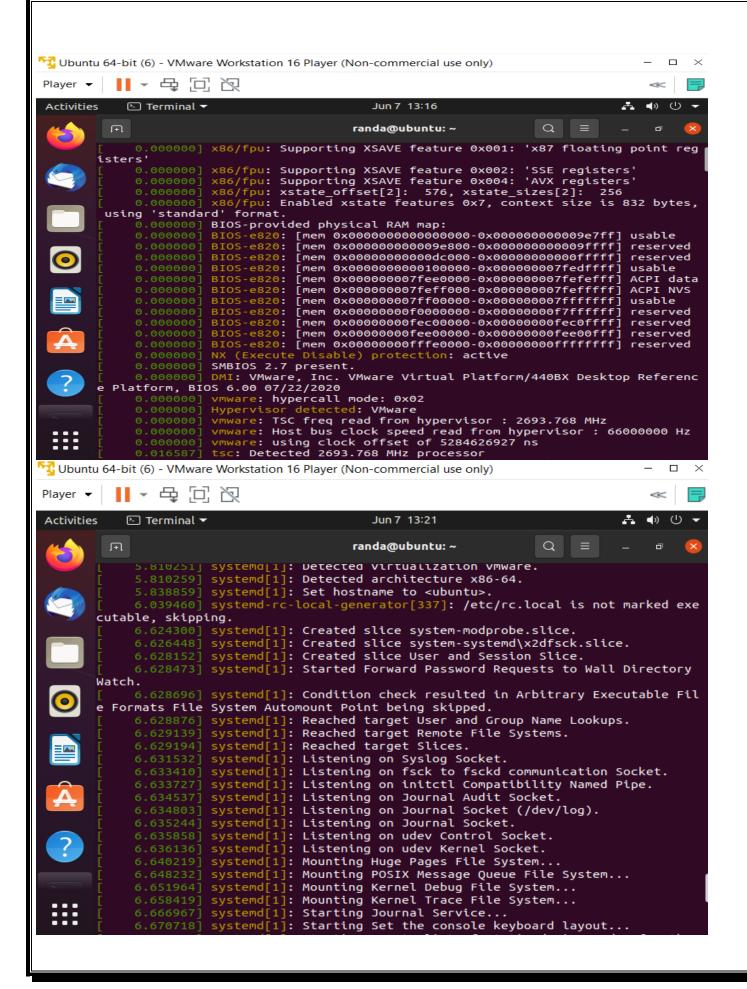
The Output

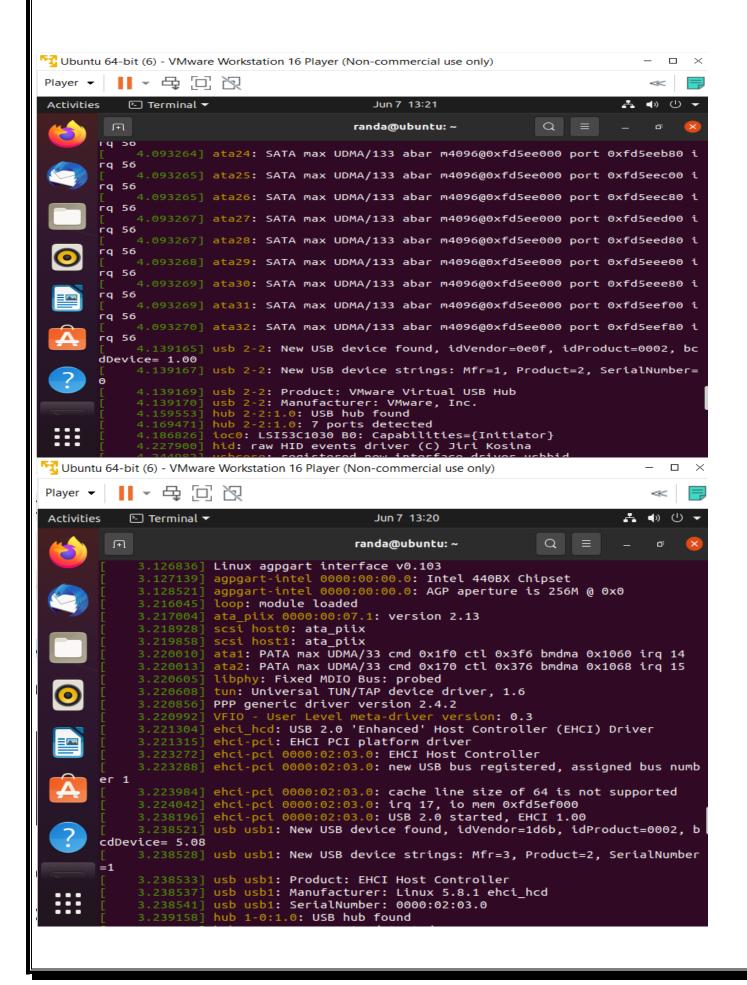
```
randa@ubuntu: ~
 Æ
randa@ubuntu:~$ gcc -o report report.c
randa@ubuntu:~$ ./report
Congratulations, mahmoud and randa!
27. Check the last line of the dmesq output.
```

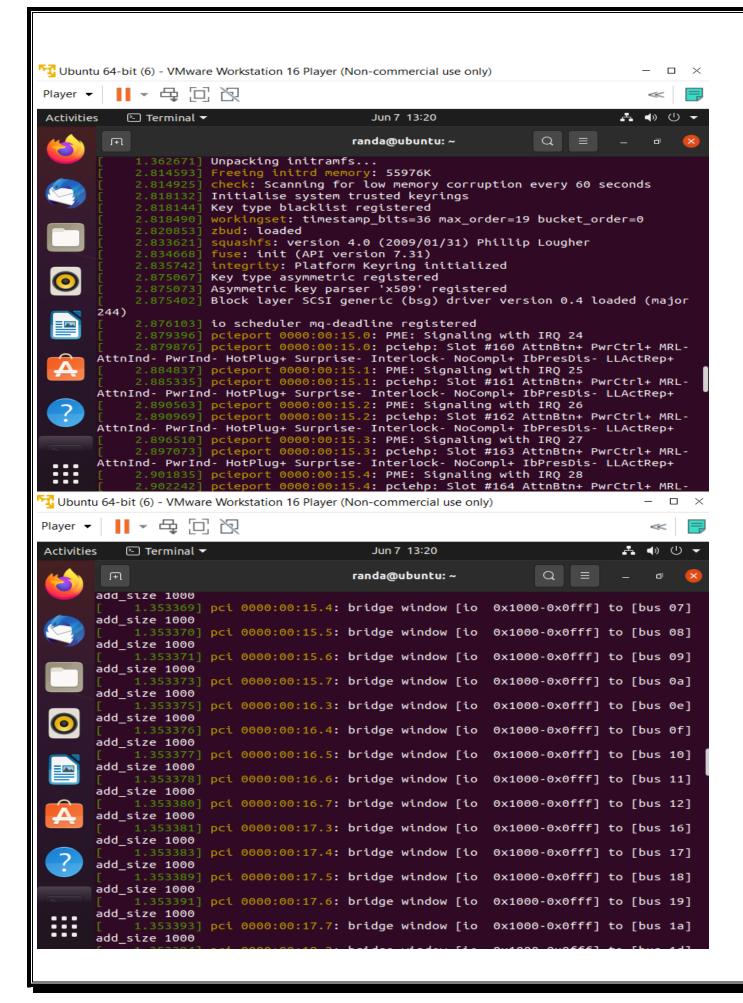
### The Output

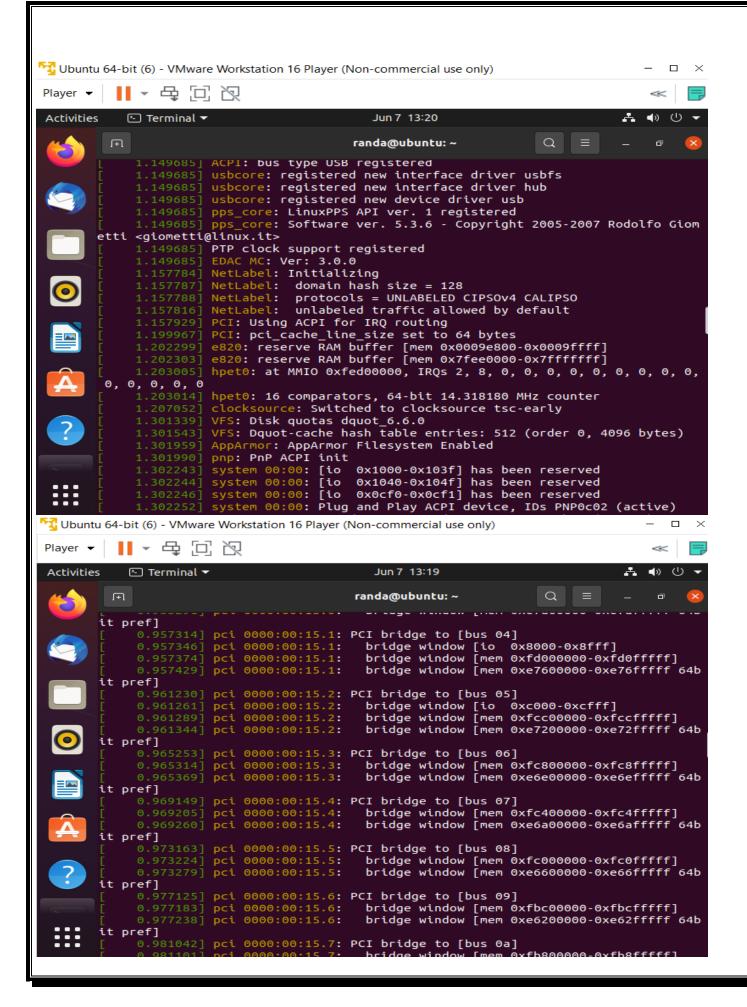
dmesg

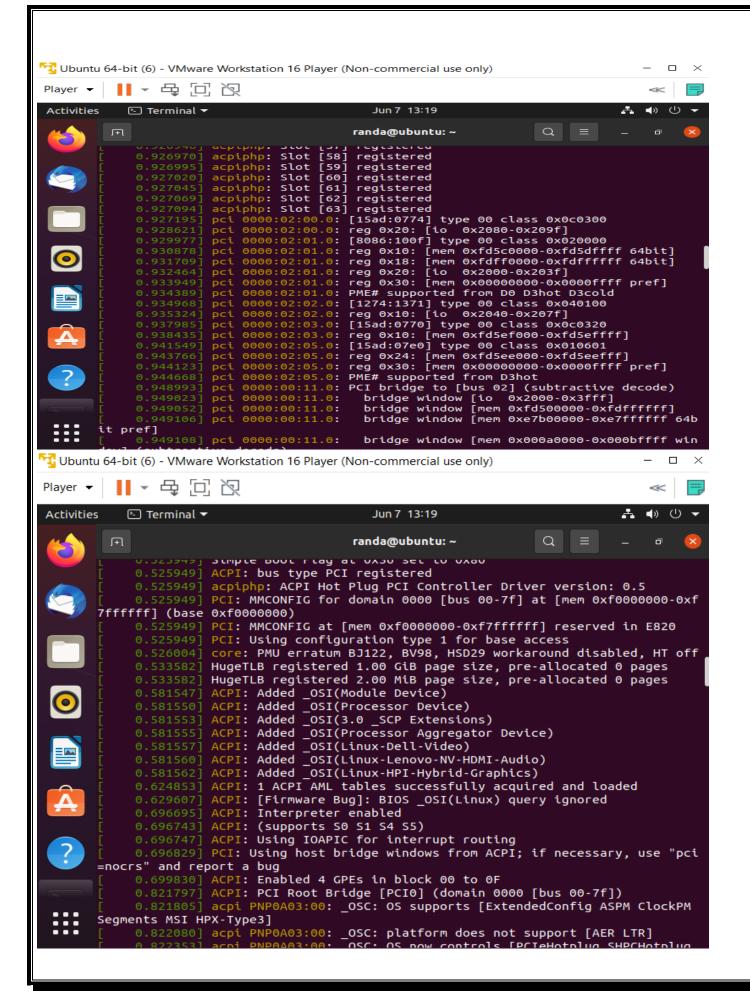


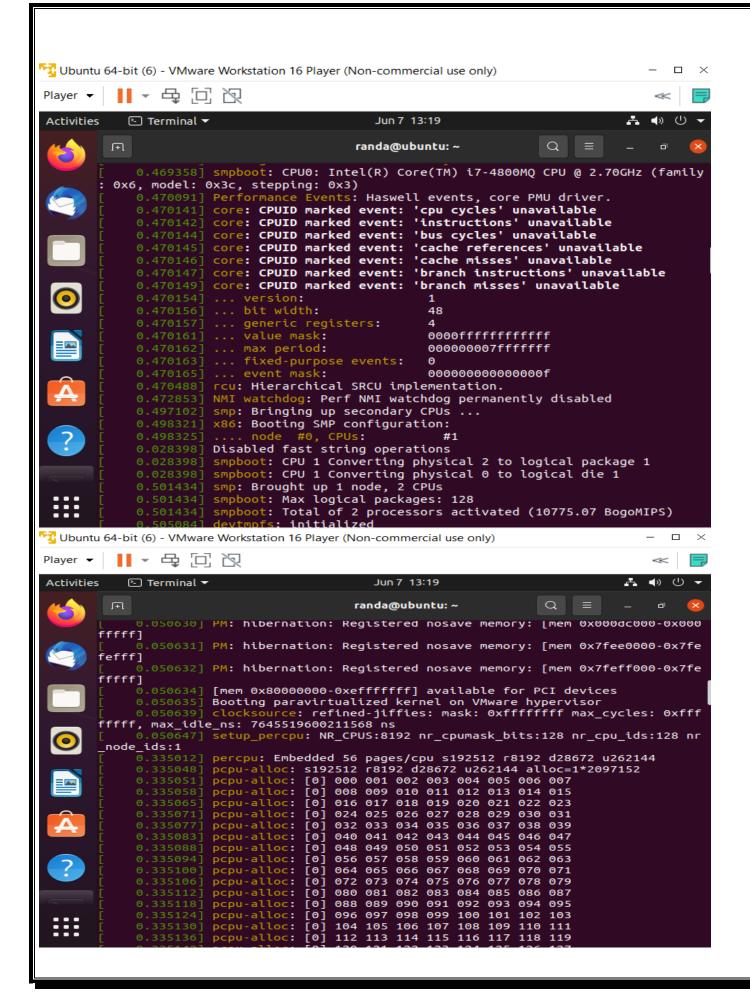


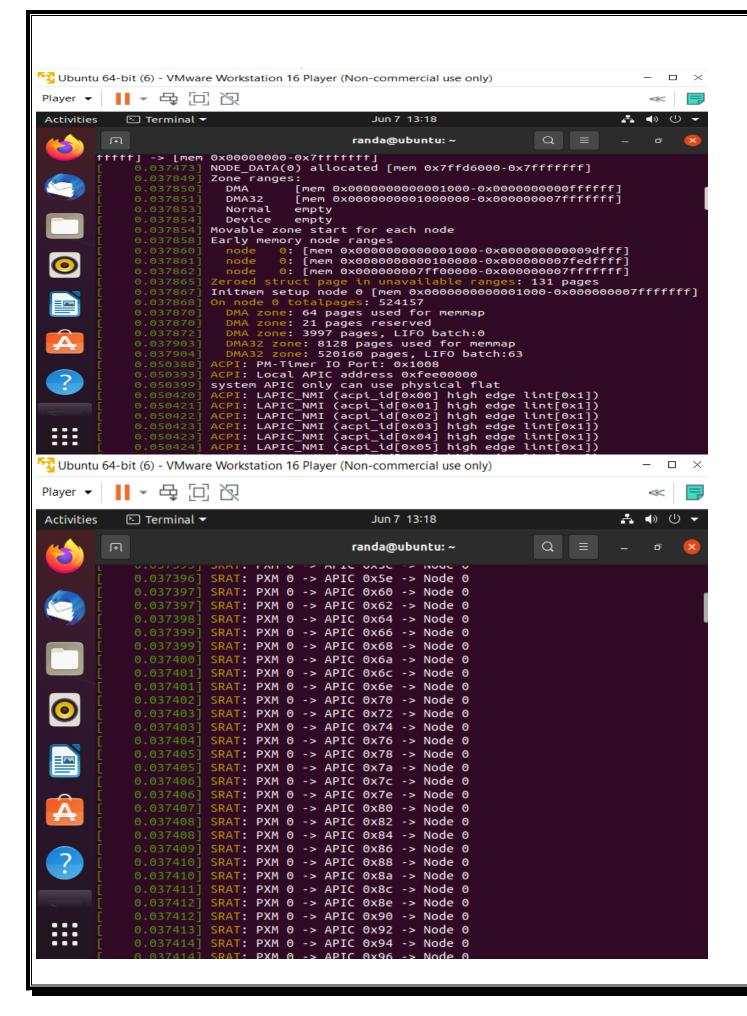


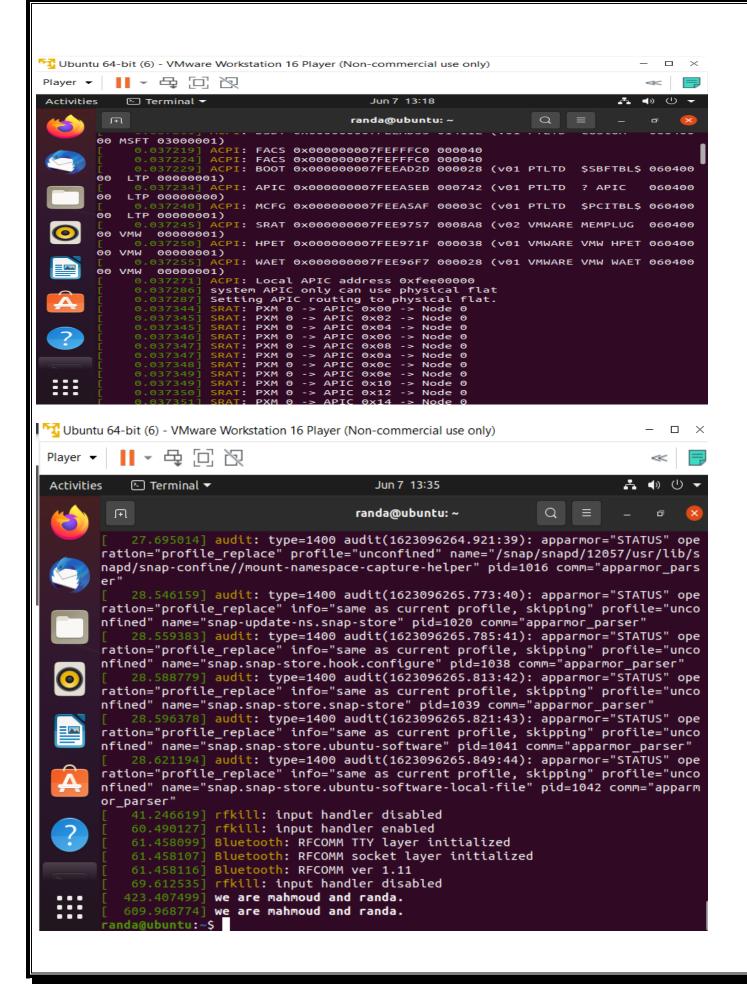












[ 423.407499] we are mahmoud and randa.
[ 609.968774] we are mahmoud and randa.
randa@ubuntu:~\$

### **References:**

 https://dev.to/jasper/adding-a-systemcall-to-the-linux-kernel-5-8-1-inubuntu-20-04-lts-2ga8