|  |
| --- |
| Linux os |
| How to add your system call |
| Ahmed Mohammed Shaban Esraa Nasser Omnia Khaild |



[Pick the date]

Final-Project-OS

First I will show the settings of my Virtual Machine :

Number of cores: 4

The capacity of memory is 4G

The kernal virsion is 5.8.1

Second:

Steps to how to add a system call :

**1.1** - Fully update your operating system by

sudo apt update && sudo apt upgrade -y

**1.2** Download and install the essential packages to compile kernels by :

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

**1.3** Clean up your installed packages.

sudo apt clean && sudo apt autoremove -y

Now Lets Go to do it :)

Now I will download the source code of the latest stable version of the Linux kernel (which is 5.8.1 as of 12 August 2020) And add i will add to my folder .

"wget -P ~/ <https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.8.1.tar.xz>" And unpack it by

**1.5** using tar -xvf ~/linux-5.8.1.tar.xz -C ~/

**1.6** Then  Reboot MY Virsual Machine

**2 -**

Now i will change to root and go on "linux-5.8.1".

Make a directory called Omnia and create file called Omnia.c in this file write a program

**#include <linux/kernel.h>**

**#include <linux/syscalls.h>**

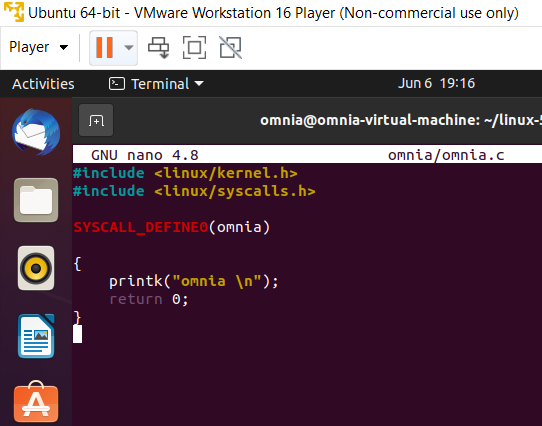
**SYSCALL\_DEFINE0(Omnia)**

**{**

**printk("Omnia.\n");**

**return 0;**

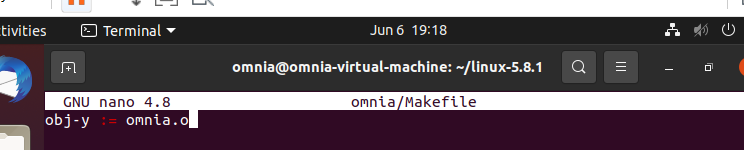
**}**

[](https://user-images.githubusercontent.com/77538165/120903019-0d3f3080-c5f0-11eb-9416-fc974766527a.png)

Save it and exit the text editor.

>> Now i will create a makefile "nano Omnia/Makefile"

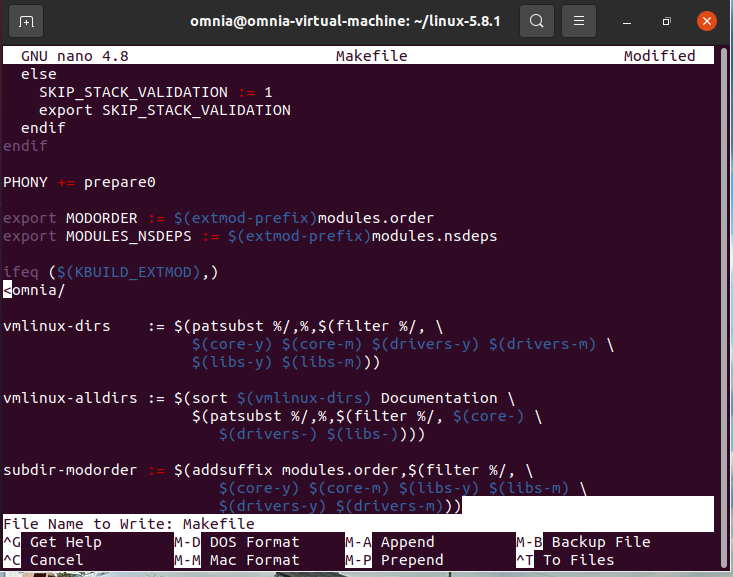
>>And write "obj-y := Omnia.o"



>>And i will open the Makefile to add the home directory to my system call to the main Makefile of the kernel.

Open the Makefile with the following command.

"nano Makefile" and i will search for core-y it will apper in the second time of searching . We did the search to see this "kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/" I will add my home directory called Omnia .

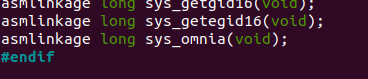
[](https://user-images.githubusercontent.com/77538165/120837357-10182380-c513-11eb-98f3-951b45b9772e.png)

And I will open the header file with the following command.

""nano include/linux/syscalls.h""

to add a corresponding function prototype for my system call to the header file of system calls.

Search for endif and put "asmlinkage long sys\_Omnia(void);" above it .

[](https://user-images.githubusercontent.com/77538165/120833899-cf1e1000-c50e-11eb-9a61-35a5c5faca42.png)

Add my system call to the kernel's system call table. By using "nano arch/x86/entry/syscalls/syscall\_64.tbl" command Note: Um 64bit if u 32bit just put 32 instead of 64.

I will navigate to the bottom of it even find a series of x32 system calls. I will put

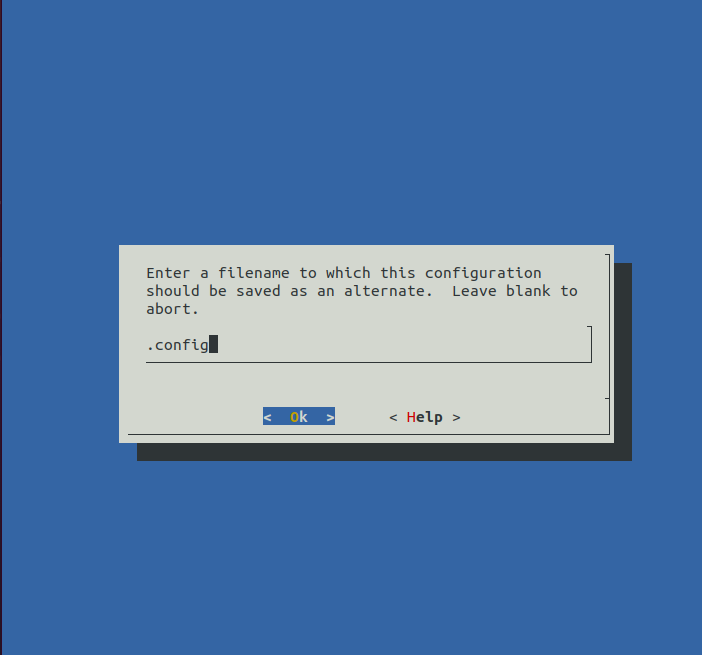
"440 common Omnia sys\_Omnia" "above the section 32 "

Now Installition:

I will install the new kernel and prepare your operating system to boot into it.

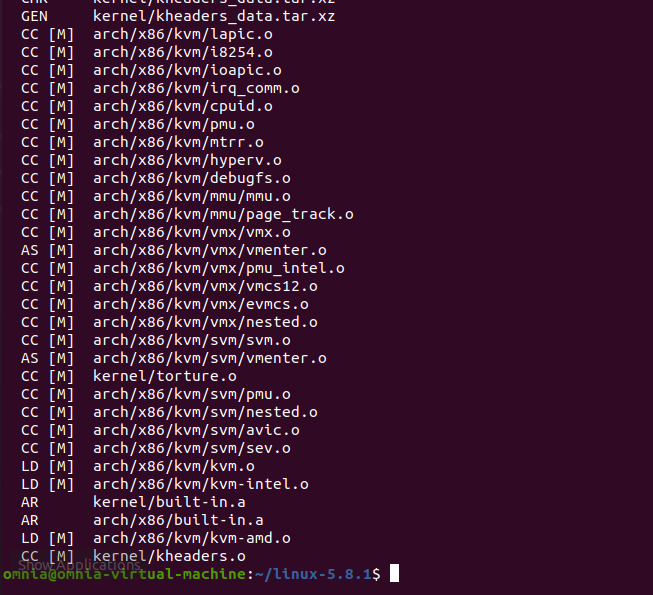
First : Configure the kernel. use "make menuconfig"

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

[](https://user-images.githubusercontent.com/77538165/120903061-6f983100-c5f0-11eb-836c-abdfaa16c8b4.png)

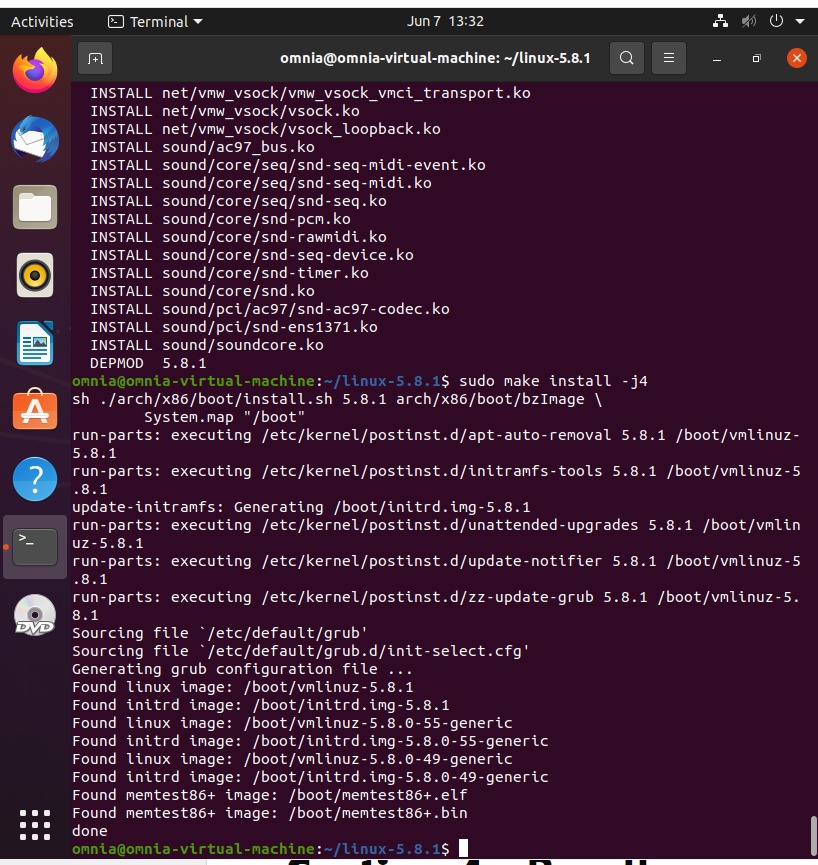
And find out how many logical cores you have. in my machine um using 4

Compile the kernel's source code. : by "make –j4"

[](https://user-images.githubusercontent.com/77538165/120834672-cf6adb00-c50f-11eb-9f23-f992649f7177.png)

Prepare the installer of the kernel.By using "sudo make modules\_install –j4"

And Install the kernel.

[](https://user-images.githubusercontent.com/77538165/120834798-fe814c80-c50f-11eb-8d81-49f45457ae99.png)

Update the bootloader of the operating system with the new kernel. by using "sudo update-grub"

Now : I will reboot my computer.

The result:

First i will change my working directory to my home directory.

Now i will Create a C file to generate a report of the success or failure of your system call.

using nano Omnia.c and put this program :

#include <linux/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 Try again .");

}

else

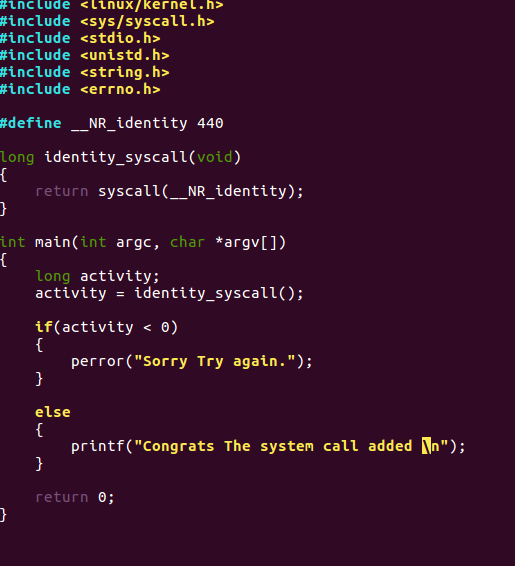
{

printf("Congratulations ,your system call added\n");

}

return 0;

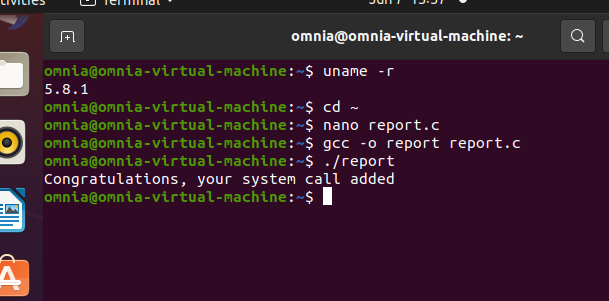
}

[](https://user-images.githubusercontent.com/77538165/120835218-810a0c00-c510-11eb-8149-4ab50b8a399e.png)

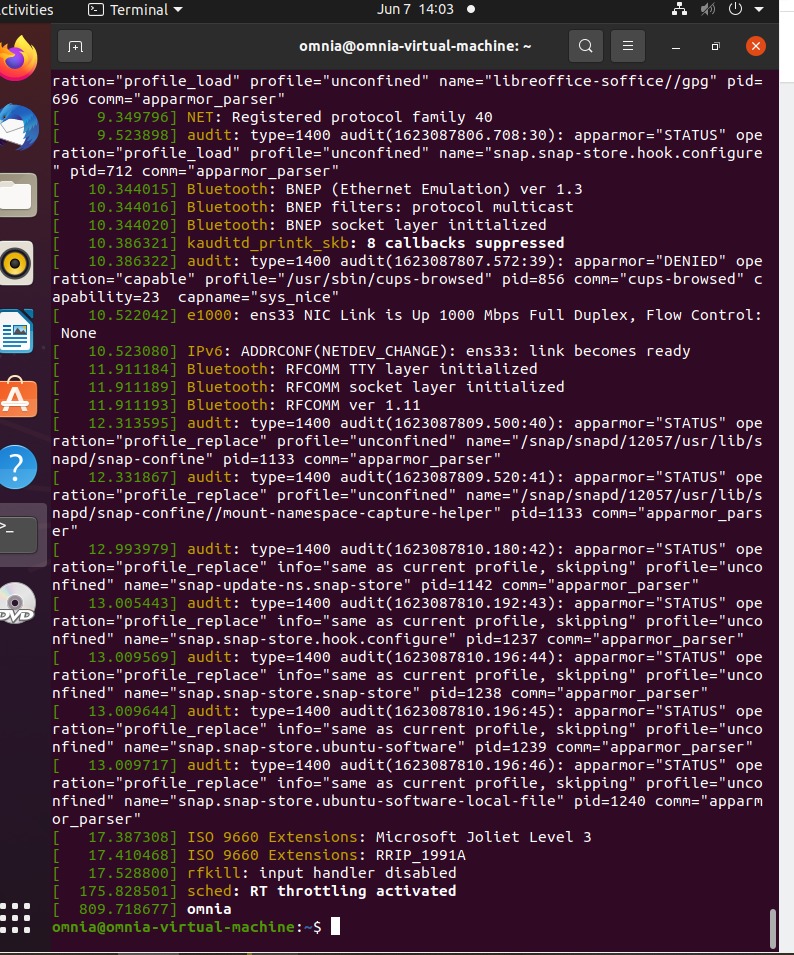
Now compile the program by usin **gcc**

**gcc -o Omnia Omnia.c**

Run and you see

[](https://user-images.githubusercontent.com/77538165/120835720-291fd500-c511-11eb-9a45-5019a24c99a3.png)

The print function :

[](https://user-images.githubusercontent.com/77538165/120903028-234cf100-c5f0-11eb-8937-15e45af9b84a.png)