Final-Project-OS

#Add System Call Called RABIX TO Kernel 5.8.1

First I will show the settings of my Virtual Machine :

Number of cores: 1

The capacity of memory is 2G

The kernal virsion is 5.8.1

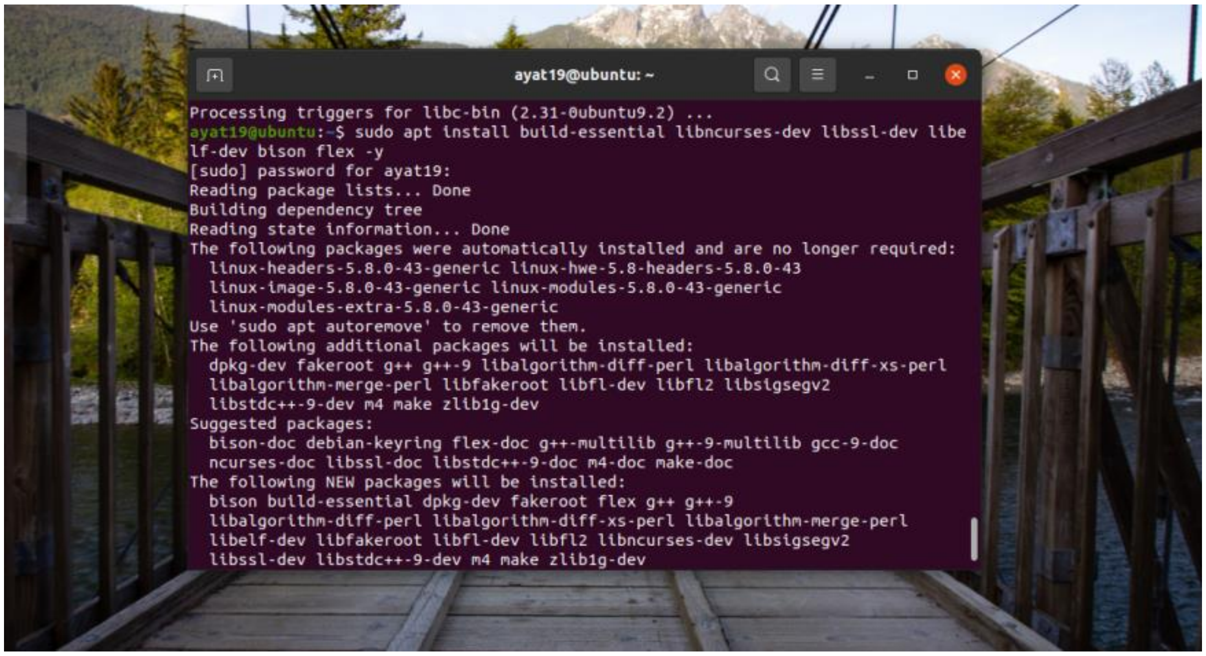
Second Steps to how to add a system call :

1- Make my Linux Ubuntu update:

```
sudo apt update && sudo apt upgrade –y
```

2- Install all packeges that i will use to compile Kernal by :

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



```
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
ayat19@ubuntu:~$ sudo apt install build-essential libncurses-dev libssl-dev libe
lf-dev bison flex -y
[sudo] password for ayat19:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-5.8.0-43-generic linux-hwe-5.8-headers-5.8.0-43
  linux-image-5.8.0-43-generic linux-modules-5.8.0-43-generic
  linux-modules-extra-5.8.0-43-generic
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  dpkg-dev fakeroot g++ g++-9 libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libfakeroot libfl-dev libfl2 libsigsegv2
  libstdc++-9-dev m4 make zlib1g-dev
Suggested packages:
  bison-doc debian-keyring flex-doc g++-multilib g++-9-multilib gcc-9-doc
  ncurses-doc libssl-doc libstdc++-9-doc m4-doc make-doc
The following NEW packages will be installed:
  bison build-essential dpkg-dev fakeroot flex g++ g++-9
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl
  libelf-dev libfakeroot libfl-dev libfl2 libncurses-dev libsigsegv2
  libssl-dev libstdc++-9-dev m4 make zlib1g-dev
```

3- Clean installed packages:

```
sudo apt clean && sudo apt autoremove –y
```

4- Download the source code of the Linux kernel 5.8.1:

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

And unpack it by using `tar -xvf ~/linux-5.8.1.tar.xz -C ~/`

5- Reboot My Computer

6- Change my working directory to the root directory of the recently unpacked source code

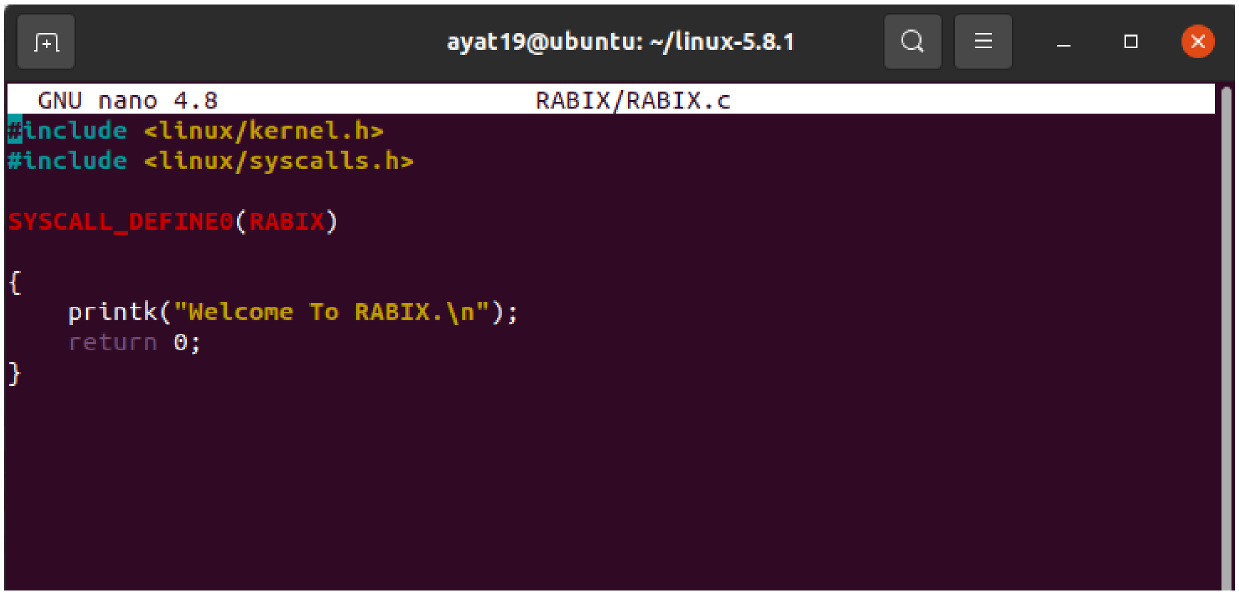
```
cd ~/linux-5.8.1/
```

7- Make a directory called RABIX and create file called RABIX.c in this file write a program

```
mkdir RABIX
```

```
nano RABIX/RABIX.c
```

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

#include <linux/syscalls.h>

SYSCALL_DEFINE0(RABIX)

{

   printk("Welcome to RABIX.\n");

   return 0;

}
```



```
  GNU nano 4.8                         RABIX/RABIX.c
#include <linux/kernel.h>
#include <linux/syscalls.h>

SYSCALL_DEFINE0(RABIX)

{
    printk("Welcome To RABIX.\n");
    return 0;
}
```

8- Now i will create a makefile

nano RABIX/Makefile

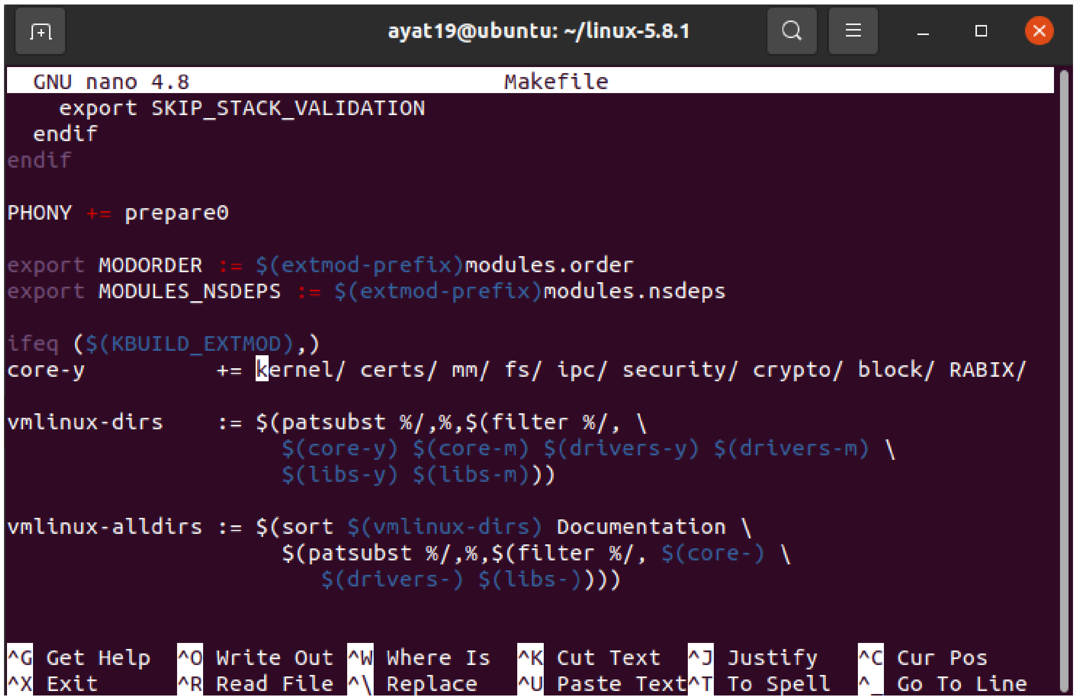
And write obj-y := RABIX.c

9- 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 RABIX .



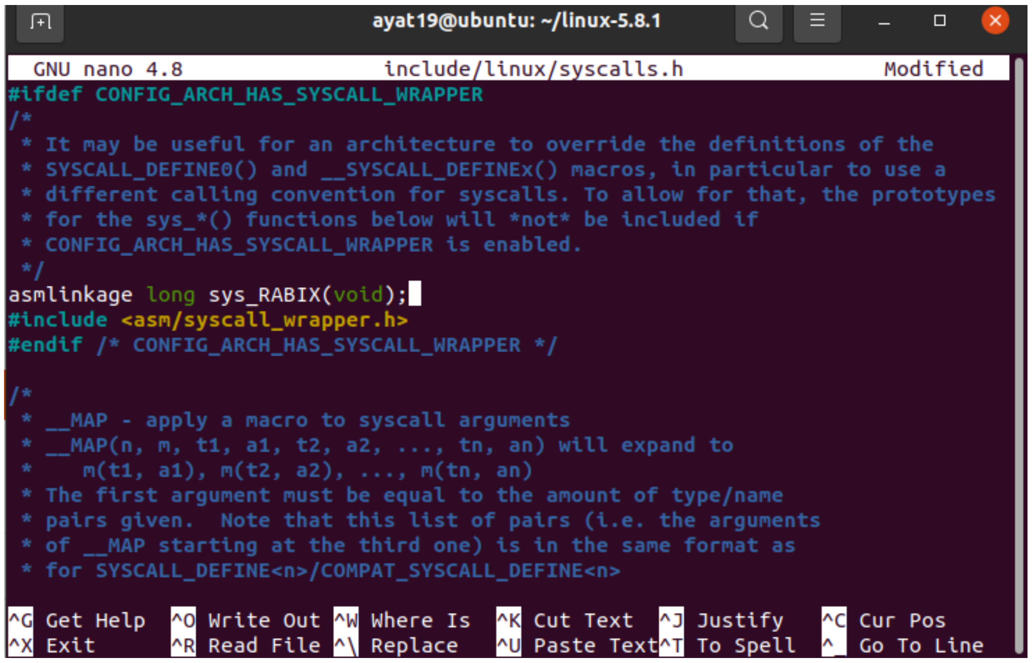
```
  GNU nano 4.8                              Makefile
    export SKIP_STACK_VALIDATION
  endif
endif

PHONY += prepare0

export MODORDER := $(extmod-prefix)modules.order
export MODULES_NSDEPS := $(extmod-prefix)modules.nsdeps

ifeq ($(KBUILD_EXTMOD),)
core-y          += kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/ RABIX/

vmlinux-dirs    := $(patsubst %/,%,$(filter %/, \
                     $(core-y) $(core-m) $(drivers-y) $(drivers-m) \
                     $(libs-y) $(libs-m)))

vmlinux-alldirs := $(sort $(vmlinux-dirs) Documentation \
                     $(patsubst %/,%,$(filter %/, $(core-) \
                        $(drivers-) $(libs-))))


^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

10- 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_RABIX(void); above it .



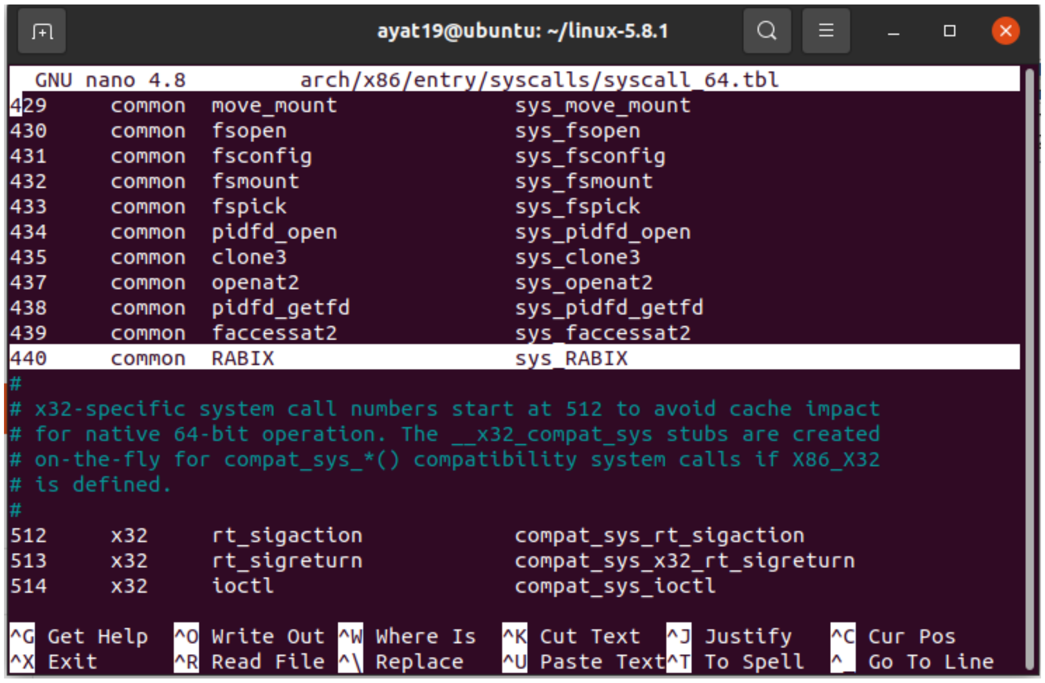
```
ayat19@ubuntu: ~/linux-5.8.1
 GNU nano 4.8               include/linux/syscalls.h                Modified
#ifdef CONFIG_ARCH_HAS_SYSCALL_WRAPPER
/*
 * It may be useful for an architecture to override the definitions of the
 * SYSCALL_DEFINE0() and __SYSCALL_DEFINEx() macros, in particular to use a
 * different calling convention for syscalls. To allow for that, the prototypes
 * for the sys_*() functions below will *not* be included if
 * CONFIG_ARCH_HAS_SYSCALL_WRAPPER is enabled.
 */
asmlinkage long sys_RABIX(void);
#include <asm/syscall_wrapper.h>
#endif /* CONFIG_ARCH_HAS_SYSCALL_WRAPPER */

/*
 * __MAP - apply a macro to syscall arguments
 * __MAP(n, m, t1, a1, t2, a2, ..., tn, an) will expand to
 *    m(t1, a1), m(t2, a2), ..., m(tn, an)
 * The first argument must be equal to the amount of type/name
 * pairs given.  Note that this list of pairs (i.e. the arguments
 * of __MAP starting at the third one) is in the same format as
 * for SYSCALL_DEFINE<n>/COMPAT_SYSCALL_DEFINE<n>
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

11- Add my system call to the kernel's system call table.

    nano arch/x86/entry/syscalls/syscall_64.tbl

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

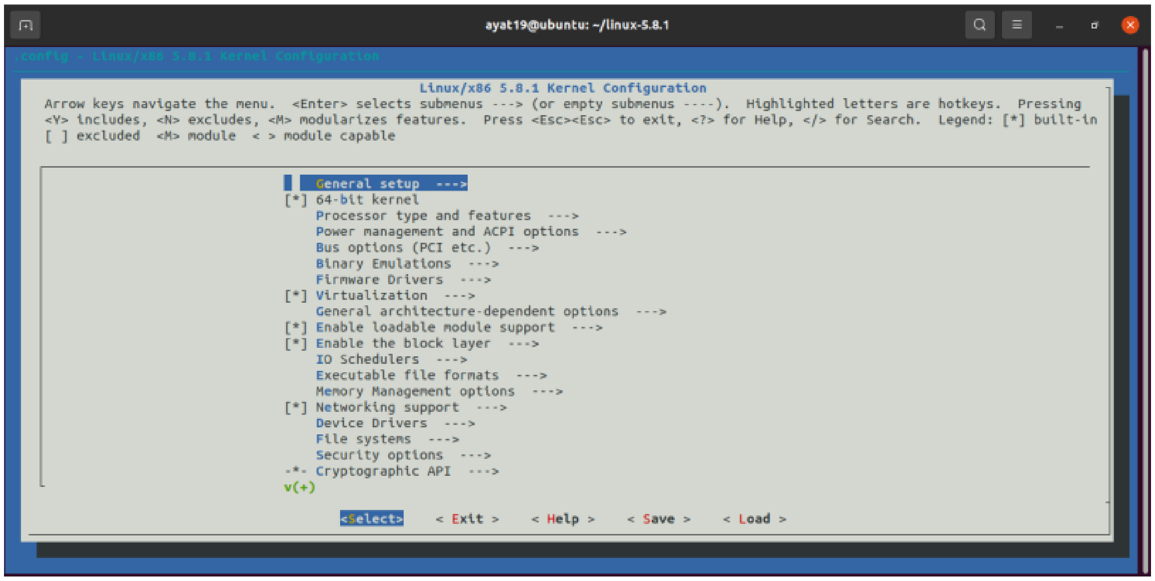
    440   common RABIX   sys_RABIX



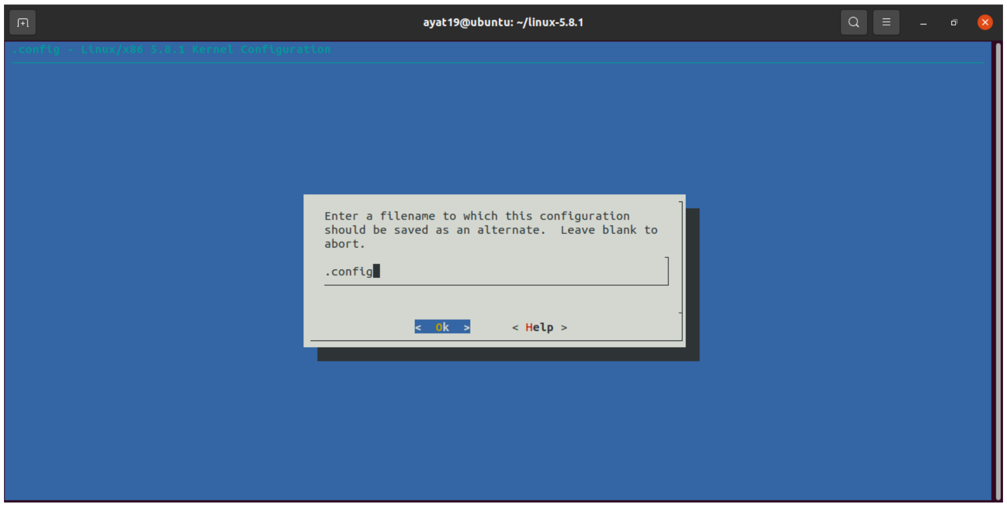
```
ayat19@ubuntu: ~/linux-5.8.1
 GNU nano 4.8          arch/x86/entry/syscalls/syscall_64.tbl
429     common  move_mount              sys_move_mount
430     common  fsopen                  sys_fsopen
431     common  fsconfig                sys_fsconfig
432     common  fsmount                 sys_fsmount
433     common  fspick                  sys_fspick
434     common  pidfd_open              sys_pidfd_open
435     common  clone3                  sys_clone3
437     common  openat2                 sys_openat2
438     common  pidfd_getfd             sys_pidfd_getfd
439     common  faccessat2              sys_faccessat2
440     common  RABIX                   sys_RABIX
#
# x32-specific system call numbers start at 512 to avoid cache impact
# for native 64-bit operation. The __x32_compat_sys stubs are created
# on-the-fly for compat_sys_*() compatibility system calls if X86_X32
# is defined.
#
512     x32     rt_sigaction            compat_sys_rt_sigaction
513     x32     rt_sigreturn            compat_sys_x32_rt_sigreturn
514     x32     ioctl                   compat_sys_ioctl
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

12- Configure the kernel.

make menuconfig



```
ayat19@ubuntu: ~/linux-5.8.1
.config - Linux/x86 5.8.1 Kernel Configuration
                         Linux/x86 5.8.1 Kernel Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable

                                  General setup  --->
                              [*] 64-bit kernel
                                  Processor type and features  --->
                                  Power management and ACPI options  --->
                                  Bus options (PCI etc.)  --->
                                  Binary Emulations  --->
                                  Firmware Drivers  --->
                              [*] Virtualization  --->
                                  General architecture-dependent options  --->
                              [*] Enable loadable module support  --->
                              [*] Enable the block layer  --->
                                  IO Schedulers  --->
                                  Executable file formats  --->
                                  Memory Management options  --->
                              [*] Networking support  --->
                                  Device Drivers  --->
                                  File systems  --->
                                  Security options  --->
                              -*- Cryptographic API  --->
                              v(+)

                  <Select>    < Exit >    < Help >    < Save >    < Load >
```



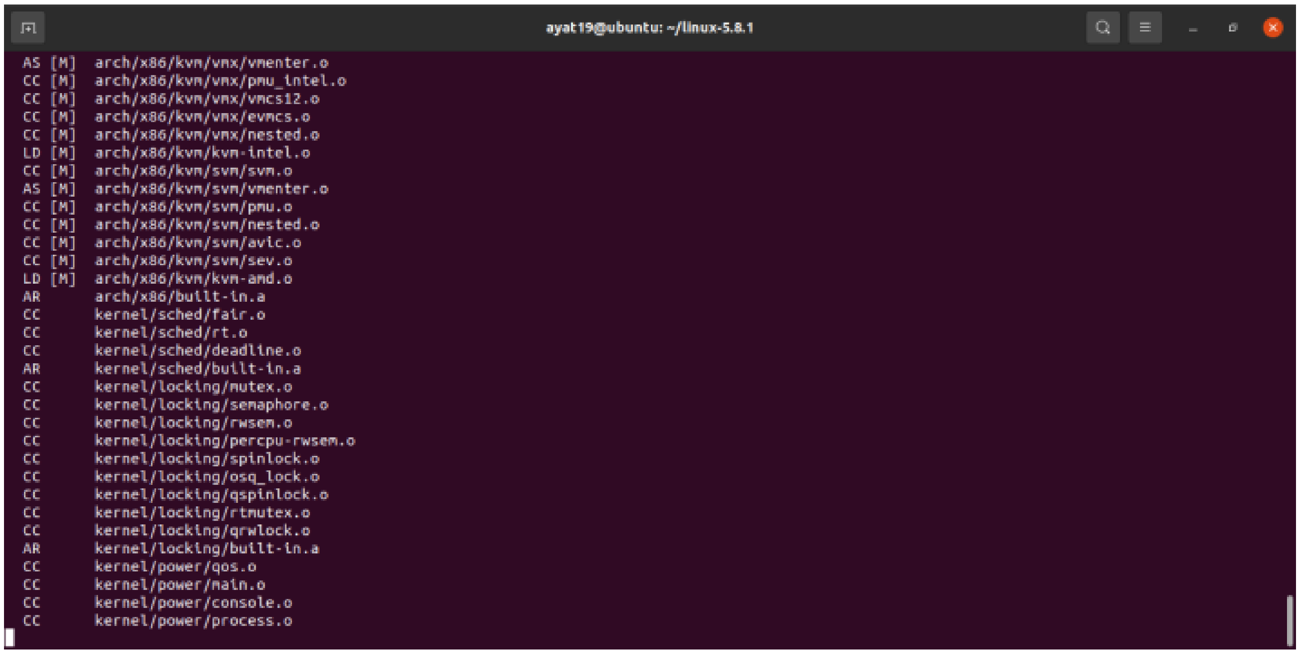
```
ayat19@ubuntu: ~/linux-5.8.1
.config - Linux/x86 5.8.1 Kernel Configuration

   Enter a filename to which this configuration
   should be saved as an alternate.  Leave blank to
   abort.

   .config

             <  Ok  >      < Help >
```

13- Find out how many logical cores you have.

Nproc

14- Compile the kernel's source code.

```
make -j1
```



```
ayat19@ubuntu: ~/linux-5.8.1
  AS [M]  arch/x86/kvm/vmx/vmenter.o
  CC [M]  arch/x86/kvm/vmx/pmu_intel.o
  CC [M]  arch/x86/kvm/vmx/vmcs12.o
  CC [M]  arch/x86/kvm/vmx/evmcs.o
  CC [M]  arch/x86/kvm/vmx/nested.o
  LD [M]  arch/x86/kvm/kvm-intel.o
  CC [M]  arch/x86/kvm/svm/svm.o
  AS [M]  arch/x86/kvm/svm/vmenter.o
  CC [M]  arch/x86/kvm/svm/pmu.o
  CC [M]  arch/x86/kvm/svm/nested.o
  CC [M]  arch/x86/kvm/svm/avic.o
  CC [M]  arch/x86/kvm/svm/sev.o
  LD [M]  arch/x86/kvm/kvm-amd.o
  AR      arch/x86/built-in.a
  CC      kernel/sched/fair.o
  CC      kernel/sched/rt.o
  CC      kernel/sched/deadline.o
  AR      kernel/sched/built-in.a
  CC      kernel/locking/mutex.o
  CC      kernel/locking/semaphore.o
  CC      kernel/locking/rwsem.o
  CC      kernel/locking/percpu-rwsem.o
  CC      kernel/locking/spinlock.o
  CC      kernel/locking/osq_lock.o
  CC      kernel/locking/qspinlock.o
  CC      kernel/locking/rtmutex.o
  CC      kernel/locking/qrwlock.o
  AR      kernel/locking/built-in.a
  CC      kernel/power/qos.o
  CC      kernel/power/main.o
  CC      kernel/power/console.o
  CC      kernel/power/process.o
```

15- Prepare the installer of the kernel.

```
sudo make modules_install -j1
```

16- Install the kernel.

```
sudo make install -j1
```

17- Update the bootloader of the operating system with the new kernel.

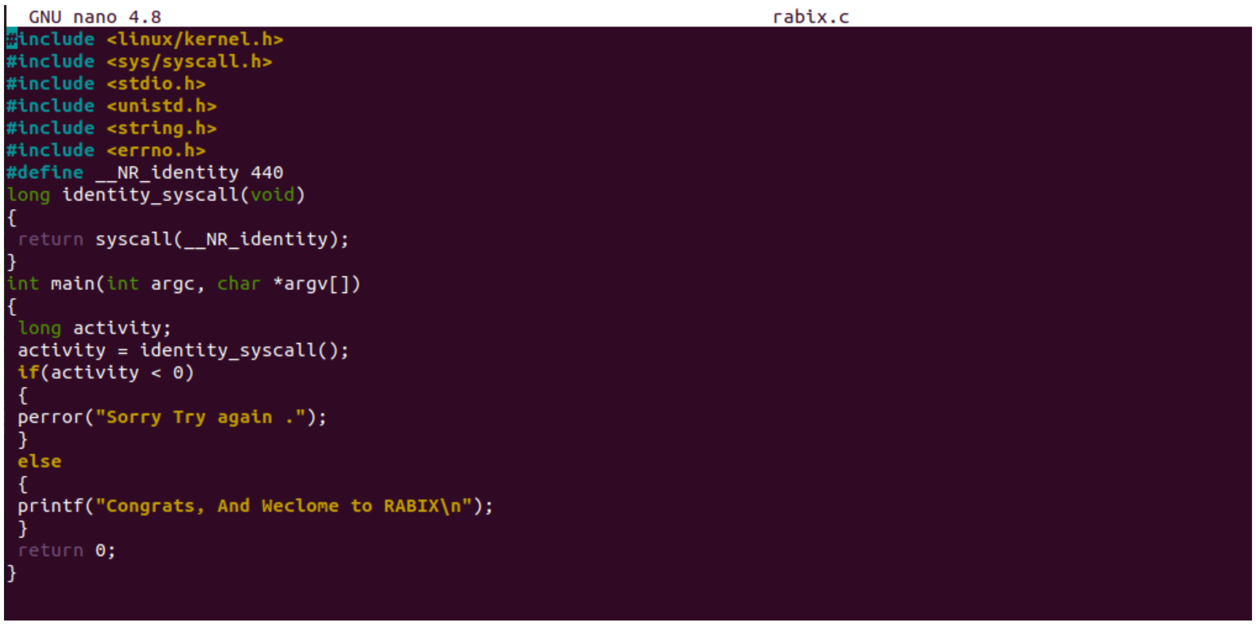
```
sudo update-grub
```

18- Reboot my computer.

19- 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 rabix.c and put this program :



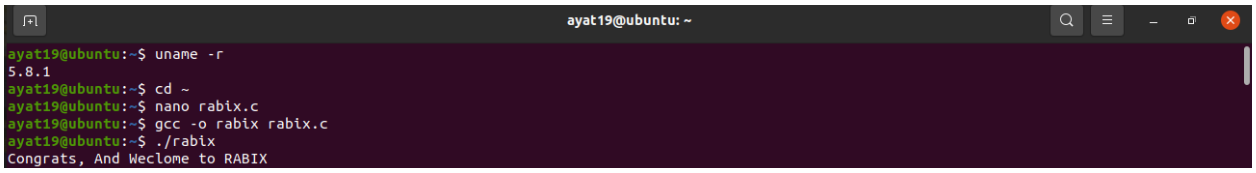
```
 GNU nano 4.8                                              rabix.c
#include <linux/kernel.h>
#include <sys/syscall.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>
#define __NR_identity 440
long identity_syscall(void)
{
 return syscall(__NR_identity);
}
int main(int argc, char *argv[])
{
 long activity;
 activity = identity_syscall();
 if(activity < 0)
 {
 perror("Sorry Try again .");
 }
 else
 {
 printf("Congrats, And Weclome to RABIX\n");
 }
 return 0;
}
```

```
#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("Congrats, And Weclome to RABIX\n");
    }
    return 0;
}
```

20- Compile the C file just created, and run C file

```
gcc -o rabix rabix.c
```

```
./rabix
```



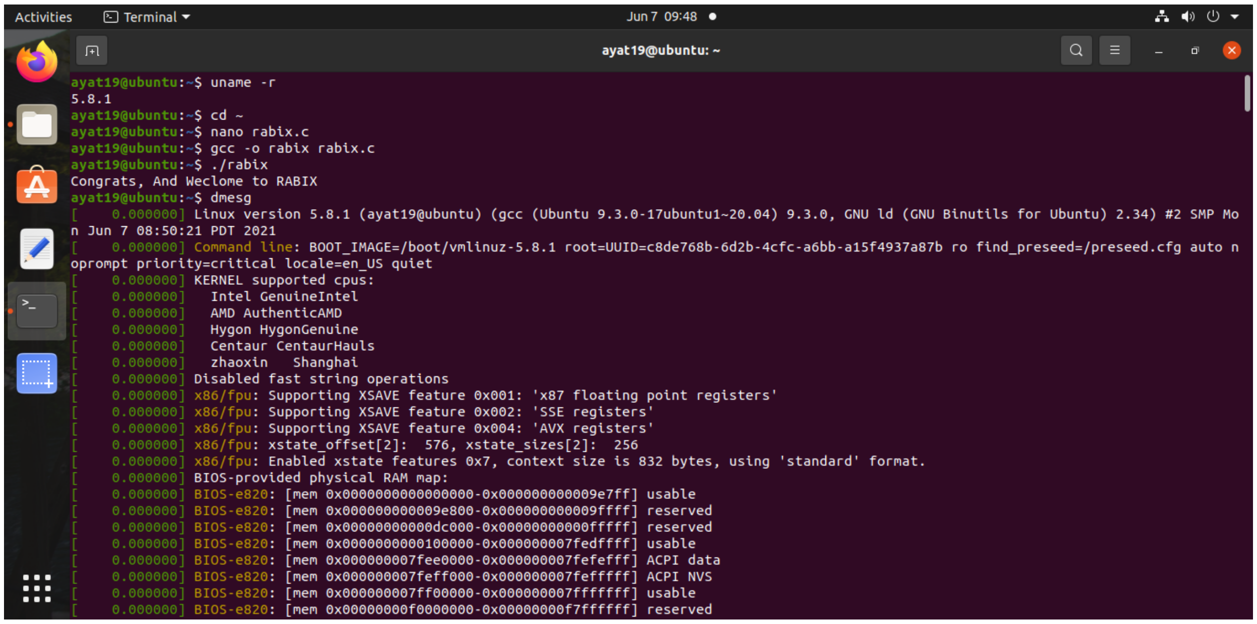
```
ayat19@ubuntu:~$ uname -r
5.8.1
ayat19@ubuntu:~$ cd ~
ayat19@ubuntu:~$ nano rabix.c
ayat19@ubuntu:~$ gcc -o rabix rabix.c
ayat19@ubuntu:~$ ./rabix
Congrats, And Weclome to RABIX
```

Will display `Congrats, And Weclome to RABIX`

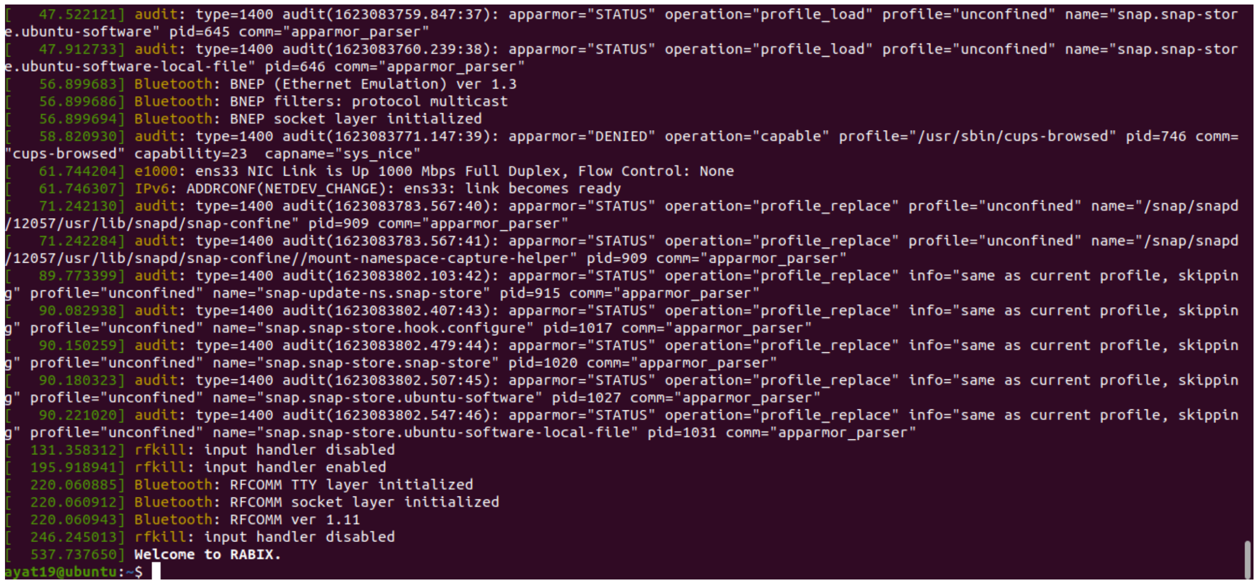
21- Check the last line of the dmesg output

The print function:

Welcome to RABIX



```
ayat19@ubuntu:~$ uname -r
5.8.1
ayat19@ubuntu:~$ cd ~
ayat19@ubuntu:~$ nano rabix.c
ayat19@ubuntu:~$ gcc -o rabix rabix.c
ayat19@ubuntu:~$ ./rabix
Congrats, And Weclome to RABIX
ayat19@ubuntu:~$ dmesg
[    0.000000] Linux version 5.8.1 (ayat19@ubuntu) (gcc (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0, GNU ld (GNU Binutils for Ubuntu) 2.34) #2 SMP Mo
n Jun 7 08:50:21 PDT 2021
[    0.000000] Command line: BOOT_IMAGE=/boot/vmlinuz-5.8.1 root=UUID=c8de768b-6d2b-4cfc-a6bb-a15f4937a87b ro find_preseed=/preseed.cfg auto n
oprompt priority=critical locale=en_US quiet
[    0.000000] KERNEL supported cpus:
[    0.000000]   Intel GenuineIntel
[    0.000000]   AMD AuthenticAMD
[    0.000000]   Hygon HygonGenuine
[    0.000000]   Centaur CentaurHauls
[    0.000000]   zhaoxin   Shanghai
[    0.000000] Disabled fast string operations
[    0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[    0.000000] x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
[    0.000000] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'standard' format.
[    0.000000] BIOS-provided physical RAM map:
[    0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009e7ff] usable
[    0.000000] BIOS-e820: [mem 0x000000000009e800-0x000000000009ffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000000dc000-0x00000000000fffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000000100000-0x000000007fedffff] usable
[    0.000000] BIOS-e820: [mem 0x000000007fee0000-0x000000007fefefff] ACPI data
[    0.000000] BIOS-e820: [mem 0x000000007feff000-0x000000007fefffff] ACPI NVS
[    0.000000] BIOS-e820: [mem 0x000000007ff00000-0x000000007fffffff] usable
[    0.000000] BIOS-e820: [mem 0x00000000f0000000-0x00000000f7ffffff] reserved
```



```
[   47.522121] audit: type=1400 audit(1623083759.847:37): apparmor="STATUS" operation="profile_load" profile="unconfined" name="snap.snap-stor
e.ubuntu-software" pid=645 comm="apparmor_parser"
[   47.912733] audit: type=1400 audit(1623083760.239:38): apparmor="STATUS" operation="profile_load" profile="unconfined" name="snap.snap-stor
e.ubuntu-software-local-file" pid=646 comm="apparmor_parser"
[   56.899683] Bluetooth: BNEP (Ethernet Emulation) ver 1.3
[   56.899686] Bluetooth: BNEP filters: protocol multicast
[   56.899694] Bluetooth: BNEP socket layer initialized
[   58.820930] audit: type=1400 audit(1623083771.147:39): apparmor="DENIED" operation="capable" profile="/usr/sbin/cups-browsed" pid=746 comm=
"cups-browsed" capability=23  capname="sys_nice"
[   61.744204] e1000: ens33 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: None
[   61.746307] IPv6: ADDRCONF(NETDEV_CHANGE): ens33: link becomes ready
[   71.242130] audit: type=1400 audit(1623083783.567:40): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/snap/snapd
/12057/usr/lib/snapd/snap-confine" pid=909 comm="apparmor_parser"
[   71.242284] audit: type=1400 audit(1623083783.567:41): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/snap/snapd
/12057/usr/lib/snapd/snap-confine//mount-namespace-capture-helper" pid=909 comm="apparmor_parser"
[   89.773399] audit: type=1400 audit(1623083802.103:42): apparmor="STATUS" operation="profile_replace" info="same as current profile, skippin
g" profile="unconfined" name="snap-update-ns.snap-store" pid=915 comm="apparmor_parser"
[   90.082938] audit: type=1400 audit(1623083802.407:43): apparmor="STATUS" operation="profile_replace" info="same as current profile, skippin
g" profile="unconfined" name="snap.snap-store.hook.configure" pid=1017 comm="apparmor_parser"
[   90.150259] audit: type=1400 audit(1623083802.479:44): apparmor="STATUS" operation="profile_replace" info="same as current profile, skippin
g" profile="unconfined" name="snap.snap-store.snap-store" pid=1020 comm="apparmor_parser"
[   90.180323] audit: type=1400 audit(1623083802.507:45): apparmor="STATUS" operation="profile_replace" info="same as current profile, skippin
g" profile="unconfined" name="snap.snap-store.ubuntu-software" pid=1027 comm="apparmor_parser"
[   90.221020] audit: type=1400 audit(1623083802.547:46): apparmor="STATUS" operation="profile_replace" info="same as current profile, skippin
g" profile="unconfined" name="snap.snap-store.ubuntu-software-local-file" pid=1031 comm="apparmor_parser"
[  131.358312] rfkill: input handler disabled
[  195.918941] rfkill: input handler enabled
[  220.060885] Bluetooth: RFCOMM TTY layer initialized
[  220.060912] Bluetooth: RFCOMM socket layer initialized
[  220.060943] Bluetooth: RFCOMM ver 1.11
[  246.245013] rfkill: input handler disabled
[  537.737650] Welcome to RABIX.
ayat19@ubuntu:~$
```

Referenece: https://dev.to/jasper/adding-a-system-call-to-the-linux-kernel-5-8-1-in-ubuntu-20-04-lts-2ga8