Project_1 Report

Group 18 E24076849 翁麒庭
E24076239 彭恩宇
E24072073 王俊傑
XX1092023 梁師睿

Implement a system call on Linux system

Part 1. Preparation

A. Install packages and update

```
root@e24076849-VirtualBox:/usr/src/linux-5.11.16# sudo apt-get install gcc libncurses5-dev bison flex libssl-dev libelf-dev make
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu gcc-9 libasan5 libatomic1 libbinutils libc-dev-bin libc6 libc6-dbg libc6-dev libcrypt-dev
  libctf-nobfd0 libctf0 libfl-dev libfl2 libgcc-9-dev libitm1 liblsan0 libncurses-dev libquadmath0 libsigsegv2 libssl1.1 libtsan0 libubsan1
  linux-libc-dev m4 manpages-dev zlib1g-dev
Suggested packages:
  binutils-doc bison-doc build-essential flex-doc gcc-multilib autoconf automake libtool gcc-doc gcc-9-multilib gcc-9-doc gcc-9-locales glibc-doc
  ncurses-doc libssl-doc m4-doc make-doc
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu bison flex gcc gcc-9 libasan5 libatomic1 libbinutils libc-dev-bin libc6-dev libcrypt-dev
  libctf-nobfd0 libctf0 libelf-dev libfl-dev libfl2 libgcc-9-dev libitm1 liblsan0 libncurses-dev libncurses5-dev libquadmath0 libsigsegv2 libssl-dev
  libtsan0 libubsan1 linux-libc-dev m4 make manpages-dev zlib1g-dev
The following packages will be upgraded:
  libc6 libc6-dbg libssl1.1
3 upgraded, 33 newly installed, 0 to remove and 95 not upgraded.
Need to get 37.0 MB/38.3 MB of archives.
After this operation, 108 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

Figure 1. Packages installed

Update: *sudo apt-get update*

Upgrade: *sudo apt-get upgrade*

B. Unpack the Linux kernel to path: *usr/src*

Command: *tar -xvf linux-5.10.33.tar.xz -C/usr/src/*

We choose the version 5.10.33

```
root@e24076849-VirtualBox:/usr/src# ls -a
.  ..  linux-5.10.33  linux-headers-5.8.0-4
```

Figure 2. Package placed in */usr/src/* folder

C. Check current system version

```
e24076849@E24076849:/usr/src/linux-5.11.16$ uname -r
5.8.0-48-generic
e24076849@E24076849:/usr/src/linux-5.11.16$
```

Figure 3. Current version of Linux (5.8.0-48)

Part 2. Creation and Detail Setting

A. Create system call file, code and *Makefile*

Makefile: Make sure the system can read *.o* file of our system call.

```
root@e24076849-VirtualBox:/usr/src/linux-5.10.33# ls -a
.               CREDITS               init         Makefile                mycall           .tmp_vmlinux.kallsyms1   virt
..              crypto                ipc          .missing-syscalls.d     net              .tmp_vmlinux.kallsyms1.o vmlinux
arch            Documentation         Kbuild       mm                      README           .tmp_vmlinux.kallsyms1.S .vmlinux.cmd
block           drivers               Kconfig      modules.builtin         samples          .tmp_vmlinux.kallsyms2   vmlinux-gdb.py
certs           fs                    kernel       modules.builtin.modinfo scripts          .tmp_vmlinux.kallsyms2.o vmlinux.o
.clang-format   .get_maintainer.ignore lib         modules.order           security         .tmp_vmlinux.kallsyms2.S vmlinux.symvers
.cocciconfig    .gitattributes        LICENSES     .modules.order.cmd      sound            tools
.config         .gitignore            .mailmap     Module.symvers          System.map       usr
COPYING         include               MAINTAINERS  .Module.symvers.cmd     .tmp_System.map  .version
```

Figure 4. Create directory of new system call (*mycall*)

```c
#include <linux/kernel.h>
#include <linux/linkage.h>
#include <linux/syscalls.h>
#include <linux/init.h>

SYSCALL_DEFINE0(mycall)
{
        printk(" This message is from the new system call.");
        return 0;
}
```

Figure 5. *mycall.c* code

```
obj-y := mycall.o
```

Figure 6. System call *Makefile*

B. Add path to core *Makefile*

Let the kernel know where to find our system call code.

```
1110
1111 ifeq ($(KBUILD_EXTMOD),)
1112 core-y          += kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/ mycall/
1113
1114 vmlinux-dirs    := $(patsubst %/,%,$(filter %/, \
1115                      $(core-y) $(core-m) $(drivers-y) $(drivers-m) \
1116                      $(libs-y) $(libs-m)))
```

Figure 7. Add path to core *Makefile*

C. Add system call to system call table

Let the system know the entry of our system call.

Since we use a 64-bit system, we edit *syscall_64.tbl* file

Path: *arch/x86/entry/syscalls/syscall_64.tbl*

```
455 439     common  faccessat2                 sys_faccessat2
456 440     common  process_madvise                    sys_process_madvise
457 441     common  epoll_pwait2                sys_epoll_pwait2
458 442     common  mycall                      sys_mycall
```

Figure 8. Add our system call to system call table

D. Add system call to system call head file

Let the system recognizes out system call.

Path: *include/linux/syscalls.h*

```
1363 int __sys_getsockopt(int fd, int level, int optname, char __user *optval,
1364                 int __user *optlen);
1365 int __sys_setsockopt(int fd, int level, int optname, char __user *optval,
1366                 int optlen);
1367 asmlinkage long sys_mycall(void);
1368 #endif
```

Figure 9. Add our system call to system call head file

Part 3. Installation

A. Generate *.config* file and setting

Command: *sudo make menuconfig*

menuconfig setting: Go into File systems, and select ext4 option, then save the configuration file as *.config*

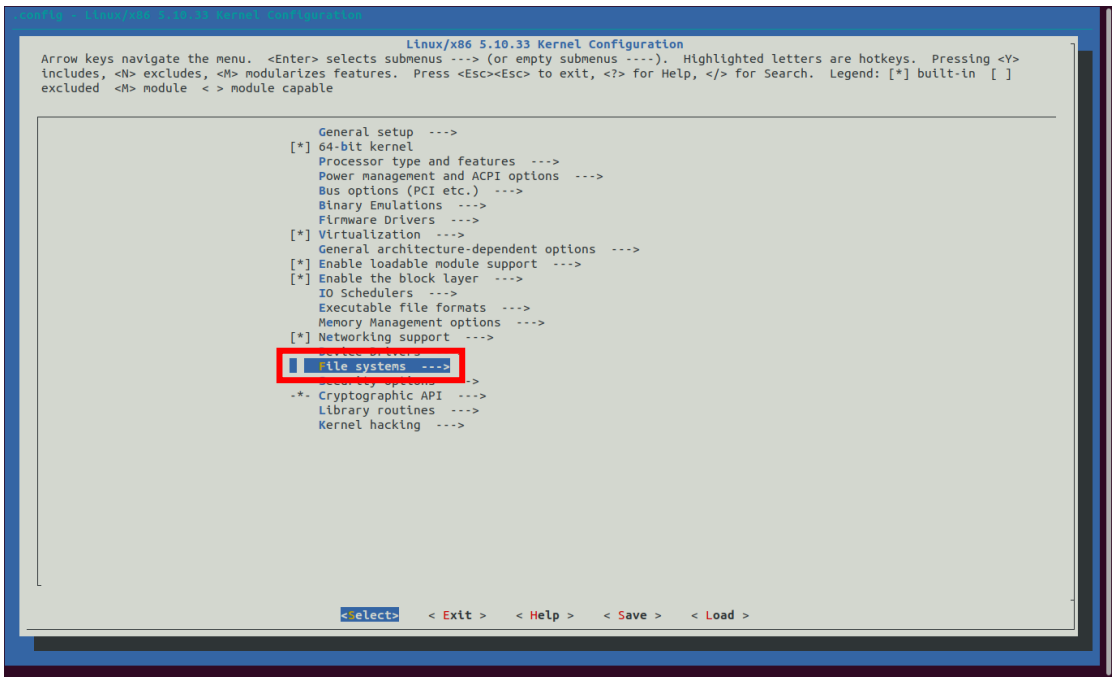

Figure 10. makeconfig window

To let the kernel compile successfully, we need to clear the *CONFIG_SYSTEM_TRUSTED_KEYS* option.

```
10469 CONFIG_MODULE_SIG_KEY="certs/signing_key.pem"
10470 CONFIG_SYSTEM_TRUSTED_KEYRING=y
10471 CONFIG_SYSTEM_TRUSTED_KEYS=""
10472 CONFIG_SYSTEM_EXTRA_CERTIFICATE=y
```

Figure 11. Clear *CONFIG_SYSTEM_TRUSTED_KEYS*

B. Compile the kernel

Command: *sudo make -j4*

Since we have four cores can use, we type in *-j4* to speed up compiling

It still took about 2 hours to finish.

An interesting thing is that our kernel end compiling without and “DONE” notification, it just stopped suddenly.

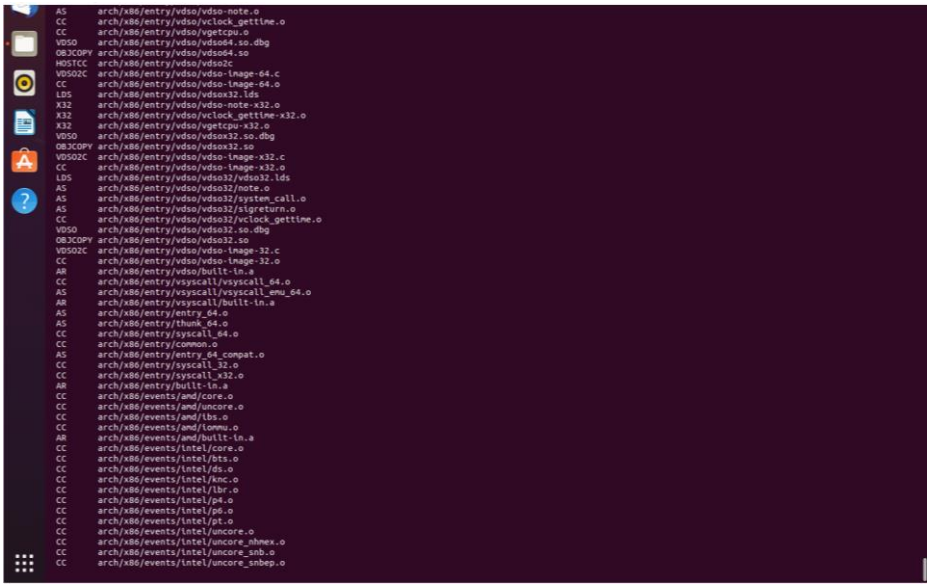

Figure 12. Kernel compiling

C. Install the kernel

Command: *sudo make modules_install install*

Install our new kernel to the system.

After install successfully, we will get four files in */boot/* directory.

config-5.10.33

initrd.img-5.10.33

System.map-5.10.33

vmlinuz-5.10.33

```
sh ./arch/x86/boot/install.sh 5.11.16 arch/x86/boot/bzImage \
        System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 5.11.16 /boot/vmlinuz-5.11.16
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 5.11.16 /boot/vmlinuz-5.11.16
update-initramfs: Generating /boot/initrd.img-5.11.16
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 5.11.16 /boot/vmlinuz-5.11.16
run-parts: executing /etc/kernel/postinst.d/update-notifier 5.11.16 /boot/vmlinuz-5.11.16
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 5.11.16 /boot/vmlinuz-5.11.16
Sourcing file `/etc/default/grub'
Sourcing file `/etc/default/grub.d/init-select.cfg'
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-5.11.16
Found initrd image: /boot/initrd.img-5.11.16
Found linux image: /boot/vmlinuz-5.8.0-50-generic
Found initrd image: /boot/initrd.img-5.8.0-50-generic
Found linux image: /boot/vmlinuz-5.8.0-43-generic
Found initrd image: /boot/initrd.img-5.8.0-43-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
done
root@e24076849-VirtualBox:/usr/src/linux-5.11.16#
```

Figure 13. Kernel installed

```
root@e24076849-VirtualBox:/boot# ls -l
total 883420
-rw-r--r-- 1 root root    250200 四  30 00:52 config-5.10.33
-rw-r--r-- 1 root root    248291 二   5 17:18 config-5.8.0-43-generic
-rw-r--r-- 1 root root    248245 四  13 02:47 config-5.8.0-50-generic
drwx------ 2 root root      4096 一   1  1970 efi
drwxr-xr-x 4 root root      4096 四  30 00:55 grub
lrwxrwxrwx 1 root root        27 四  29 21:50 initrd.img -> initrd.img-5.8.0-50-generic
-rw-r--r-- 1 root root 752018838 四  30 00:55 initrd.img-5.10.33
-rw-r--r-- 1 root root  52051811 四  29 22:43 initrd.img-5.8.0-43-generic
-rw-r--r-- 1 root root  52906685 四  29 22:44 initrd.img-5.8.0-50-generic
lrwxrwxrwx 1 root root        27 四  29 21:39 initrd.img.old -> initrd.img-5.8.0-43-generic
-rw-r--r-- 1 root root    182704 八  18  2020 memtest86+.bin
-rw-r--r-- 1 root root    184380 八  18  2020 memtest86+.elf
-rw-r--r-- 1 root root    184884 八  18  2020 memtest86+_multiboot.bin
-rw-r--r-- 1 root root   5749228 四  30 00:52 System.map-5.10.33
-rw------- 1 root root   5515823 二   5 17:18 System.map-5.8.0-43-generic
-rw------- 1 root root   5531453 四  13 02:47 System.map-5.8.0-50-generic
lrwxrwxrwx 1 root root        15 四  30 00:52 vmlinuz -> vmlinuz-5.10.33
-rw-r--r-- 1 root root  10001440 四  30 00:52 vmlinuz-5.10.33
-rw-r--r-- 1 root root   9716672 二  10 03:04 vmlinuz-5.8.0-43-generic
-rw------- 1 root root   9785696 四  13 04:17 vmlinuz-5.8.0-50-generic
lrwxrwxrwx 1 root root        24 四  29 21:50 vmlinuz.old -> vmlinuz-5.8.0-50-generic
```

Figure 14. Four files generated after install successfully

D. Update system and reboot

Command: *sudo update-grub*

Make sure we update the kernel version, then we reboot the system.

Part 4. Testing and Result

A. Check system version

Command: *uname -r*

Our system version has been updated to 5.10.33

```
e24076849@e24076849-VirtualBox:~$ uname -r
5.10.33
e24076849@e24076849-VirtualBox:~$
```

Figure 15. New kernel implemented successfully

B. Write a user level code to call our system call

```c
#include <linux/kernel.h>
#include <sys/syscall.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>

long call(void)
{
        return syscall(441);
}

int main(int argc, char *argv[])
{
        long activity;
        activity = call();

        if (syscall < 0)
        {
                printf("System call failed\n");
        }
        else
        {
                printf("System call success, return %ld\n", activity);
        }
        return 0;
}
```

Figure 16. *call.c* code (user level code)

C. Check our system call output

After compile user level code and run it, we get the result from both code and system call.

Command: *dmesg* To check system call output

```
root@e24076849-VirtualBox:/home/e24076849# gcc -o call call.c
root@e24076849-VirtualBox:/home/e24076849# ./call
System call success, return 0
```

Figure 17. user level code result

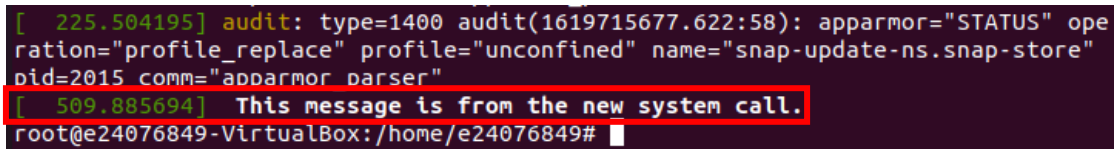

Figure 17. System call output

Part 5. Review and Experience

Question we met:

1. We didn't meet any problems in Part A and Part B, we type in and modify the files according to the tutorial online.
2. Almost our problems occurs when we compile the kernel, Part C of the steps.

   First, we encounter a pass key acquired whiling compiling. We remove this by clear the *CONFIG_SYSTEM_TRUSTED_KEYS* setting, let the kernel acquire no key to compile.

   Next, the largest problem we met is *undefine reference*.

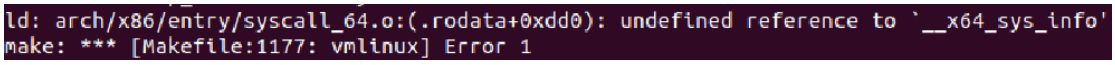

Figure 18. Undefine reference error

(*our call is named *sys_info* instead of *sys_mycall* in this try)

We notice that '*__x64_*' may automatically added to our system call table while compiling, let the system can't recognize it.

As the solution found online, one told that we should modify *arch/arm/tools/syscall.tbl* and *kernel/Makefile* instead of *arch/x86/entry/syscalls/syscall_64.tbl*

But *no rule to make target 'mycall.o'* error occurs, thus we give up this way modifying the kernel files.

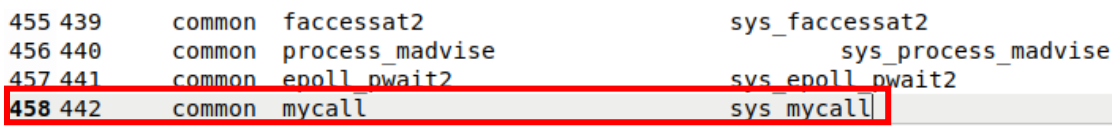

Figure 19. Modify in *arch/arm/tools/syscall.tbl*

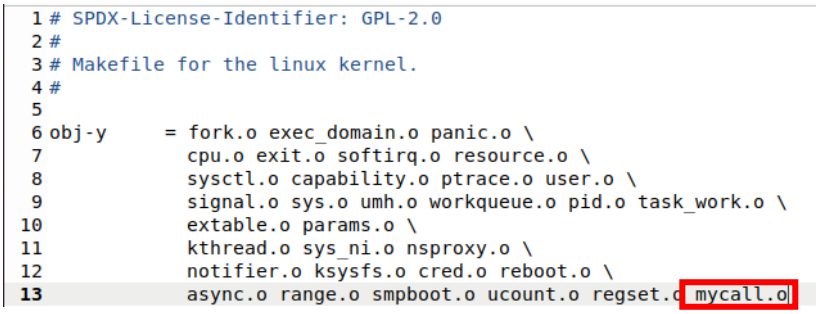

Figure 20. Modify in *kernel/Makefile*

Another online tutorial said we should modify both *syscall_32.tbl* and *syscall_64.tbl* for the system to recognize our system call, but we encounter the same undefine error.

Then we tried to modify *syscall_64.tbl* as in the figure, but then we get undefine reference '*__x64___x64_sys_info*'

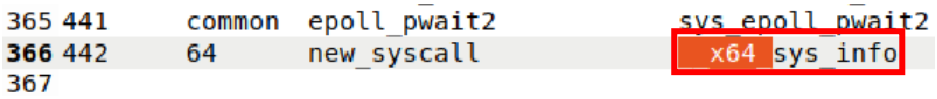

Figure 21. Another trying to modify *syscall_64.tbl*

Last, we select to change our modifying kernel version from 5.11.16 to 5.10.33, then everything goes perfect.

We change our system call from *sys_info* to *sys_mycall* due to we thought '*sys_info*' this name may be too similar to system calls implemented in the kernel already.

3. During compiling, we encounter a fatal error: *not enough space*, thus we extend our virtual box storage from 32GB to 128GB.

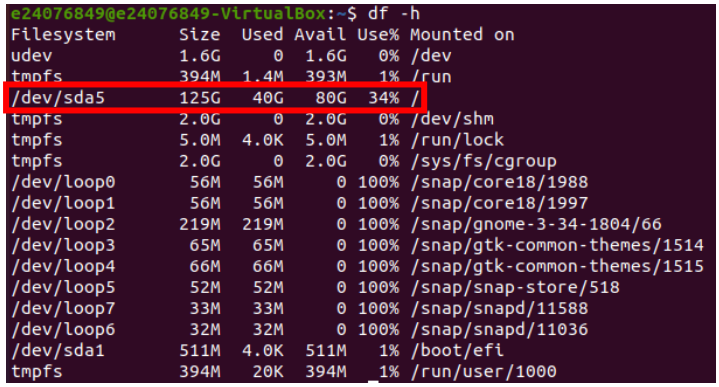

Figure 22. Storage used up after installing

Experience:

After a lot of research and discussing, we finally implement a simple system call. The tutorials online have many bugs which they didn't wrote in their post. We put everything together in about 5 tutorials to let our system call work properly.

During the experiment, we now know how Linux system system call work.

Construct a system call C code and a *Makefile* for system to read, then add the path of core *Makefile* let kernel to know where to find our system call, last add our system call to system call table and head file for kernel to know the entry and codename when we call our system call through a user level code.

Compiling the kernel is also a big problem. Many errors may occur in different version after we modify some files. The most confusing part is the *undefine reference* error, it automatically adds a stub to our system call but can't recognize it. This bug took we about 5 days to solve.

Division of work:

E24076849 翁麒庭:

Researching, Debugging (Main), Coding

E24076239 彭恩宇:

Researching, Debugging, Coding, Report writing

E24072073 王俊傑:

Researching, Debugging, Report writing (Main)

XX1092023 梁師睿:

Didn't shown up

Part 6. File hierarchy

call.c	- User level code to call system call
linux-5.10.33	- Kernel file
mycall	- System call folder
mycall.c	- System call C code
Makefile	- System call Makefile
include/linux/	- Head file folder
syscalls.h	- System call head file
arch/x86/entry/syscalls/	- System call table folder
syscall_64.tbl	- System call table for 64-bit
Makefile	- Core Makefile
config	- Kernel configuration fiel

Part 7. Reference:

http://linux.vbird.org/linux_basic/0540kernel.php

https://medium.com/anubhav-shrimal/adding-a-hello-world-system-call-to-linux-kernel-dad32875872

https://hackmd.io/@combo-tw/Linux-%E8%AE%80%E6%9B%B8%E6%9C%83/%2F%40combo-tw%2FBJPoAcqQS

http://chriswenyuan.blogspot.com/2017/05/system-call-linux-kernel-v4x.html

https://stackoverflow.com/questions/26720644/adding-new-system-call-to-linux-kernel-3-13-on-64-bit-system

https://finonglager2145.pixnet.net/blog/post/63664471

http://www.cjwind.idv.tw/Add-system-call-to-linux/

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

https://stackoverflow.com/questions/64798216/adding-a-new-system-call-linux-5-9-8-and-make-it-appear-on-boot-screen

https://home.gamer.com.tw/creationDetail.php?sn=2873436