Linux System Call

Practical Class 5

Systems and Storage Laboratory
Department of Computer Science and Engineering
Chung-Ang University

Index

- **Creating a new system call**
  1. Download Kernel
  2. Define system call function
  3. Register system call number to the table
  4. Register system call to the header file
  5. Kernel compile & change
  6. System call usage

Download Kernel (Practical class 4)

- **Check current kernel version**
  - `$ uname -r`

- **Download kernel code from https://www.kernel.org/**
- **Or use following command:**

```
$ wget https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/linux-5.x.tar.gz
$ tar -xvf linux-5.x.tar.gz
```

Define system call function

- Create system call function
  - $ vim linux-x.x.x/kernel/my_syscall.c (Make new file)

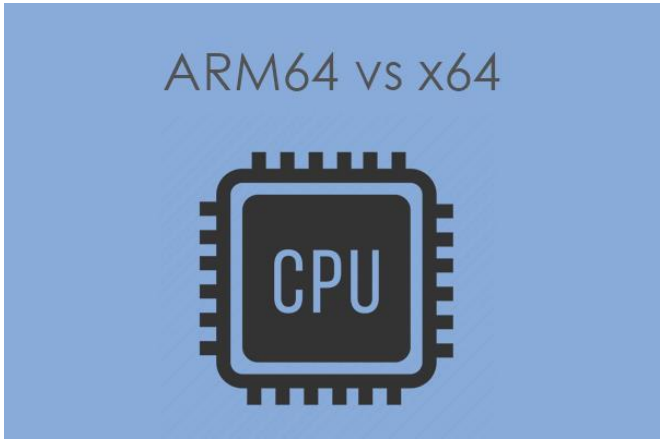
```
1 #include <linux/syscalls.h>
2
3 SYSCALL_DEFINE0(mycall)
4 {
5         printk("System Call Example!\n");
6
7         return 0;
8 }
```

- Add this code to Makefile
  - $ vim linux-x.x.x/kernel/Makefile

```
obj-y     = fork.o exec_domain.o panic.o \
            cpu.o exit.o softirq.o resource.o \
            sysctl.o sysctl_binary.o capability.o ptrace.o user.o \
            signal.o sys.o umh.o workqueue.o pid.o task_work.o \
            extable.o params.o \
            kthread.o sys_ni.o nsproxy.o \
            notifier.o ksysfs.o cred.o reboot.o \
            async.o range.o smpboot.o ucount.o my_syscall.o
```

Register system call number to the table

- We need to add the system call number to the table.
- There are 2 popular architecture:
  - X86: Intel Processor
  - Arm64: Apple M1, M2, some AMD, smart tablet processors
- The system call table is in different files based on the architecture. You should check the processor architecture before implement the next step.



Register system call number to the table

- x86
  - $ vim linux-x.x.x/arch/x86/entry/syscalls/syscall_64.tbl

```
...
<NUM>     common    mycall                  __x64_sys_mycall   # add syscall number

...
```

```
424     common  pidfd_send_signal       __x64_sys_pidfd_send_signal
425     common  io_uring_setup          __x64_sys_io_uring_setup
426     common  io_uring_enter          __x64_sys_io_uring_enter
427     common  io_uring_register       __x64_sys_io_uring_register
428     common  open_tree               __x64_sys_open_tree
429     common  move_mount              __x64_sys_move_mount
430     common  fsopen                  __x64_sys_fsopen
431     common  fsconfig                __x64_sys_fsconfig
432     common  fsmount                 __x64_sys_fsmount
433     common  fspick                  __x64_sys_fspick
434     common  pidfd_open              __x64_sys_pidfd_open
435     common  clone3                  __x64_sys_clone3/ptregs
```

Before

```
424     common  pidfd_send_signal       __x64_sys_pidfd_send_signal
425     common  io_uring_setup          __x64_sys_io_uring_setup
426     common  io_uring_enter          __x64_sys_io_uring_enter
427     common  io_uring_register       __x64_sys_io_uring_register
428     common  open_tree               __x64_sys_open_tree
429     common  move_mount              __x64_sys_move_mount
430     common  fsopen                  __x64_sys_fsopen
431     common  fsconfig                __x64_sys_fsconfig
432     common  fsmount                 __x64_sys_fsmount
433     common  fspick                  __x64_sys_fspick
434     common  pidfd_open              __x64_sys_pidfd_open
435     common  clone3                  __x64_sys_clone3/ptregs
436     common  mycall                  __x64_sys_mycall
```

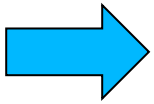
After

Register system call number to the table

- **Arm64**
  - `$ vim linux-x.x.x/include/uapi/asm-generic/unistd.h`
- **Use 460 for the mycall number (if you have error, increase higher)**
- **__NR_syscalls should be 461 (mycall +1)**

```
#ifdef __ARCH_WANT_SYS_CLONE3
#define __NR_clone3 435
__SYSCALL(__NR_clone3, sys_clone3)
#endif

#undef __NR_syscalls
#define __NR_syscalls 436

/*
 * 32 bit systems traditionally used different
 * syscalls for off_t and loff_t arguments, whi
```

Before



```
#ifdef __ARCH_WANT_SYS_CLONE3
#define __NR_clone3 435
__SYSCALL(__NR_clone3, sys_clone3)
#endif

#define __NR_mycall 460
__SYSCALL(__NR_mycall, sys_mycall)

#undef __NR_syscalls
#define __NR_syscalls 461
```

After

Register system call to header file

- **Add our system call to syscalls.h header file**
  - **`$ vim linux-x.x.x/include/linux/syscalls.h`**
  - Add our system call function

```
...
asmlinkage long sys_mycall(void);
#endif /* CONFIG_ARCH_HAS_SYSCALL_WRAPPER */
```

```
1219 /*
1220  * Not a real system call, but a placeholder for syscalls which are
1221  * not implemented -- see kernel/sys_ni.c
1222  */
1223 asmlinkage long sys_ni_syscall(void);
1224
1225 asmlinkage long sys_mycall(void);
1226
1227 #endif /* CONFIG_ARCH_HAS_SYSCALL_WRAPPER */
```

Kernel compile

- Kernel compile
  - `$ make menuconfig (if it is not configured)`
  - `$ make -j 4`
  - `$ make modules -j 4`
  - `$ sudo make modules_install -j 4`
  - `$ sudo make install`

Change the default boot option

- *If the grub is not configured for the new compiled kernel, you can follow the below steps:*
- **Check grub id from there**
  - `$ vim /boot/grub/grub.cfg`
- **Change grub_default setting**
  - `$ vim /etc/default/grub`

```
GRUB_DEFAULT=“submenu_id>menuentry_id”

Ex)
GRUB_DEFAULT="gnulinux-advanced-65c9af03-3d9b-411c-99b2-a9ada0961a40>gnulinux-
4.7.0-1-amd64-advanced-65c9af03-3d9b-411c-99b2-a9ada0961a4"
```

- **Apply grub configuration**
  - `$ update-grub`

System call usage

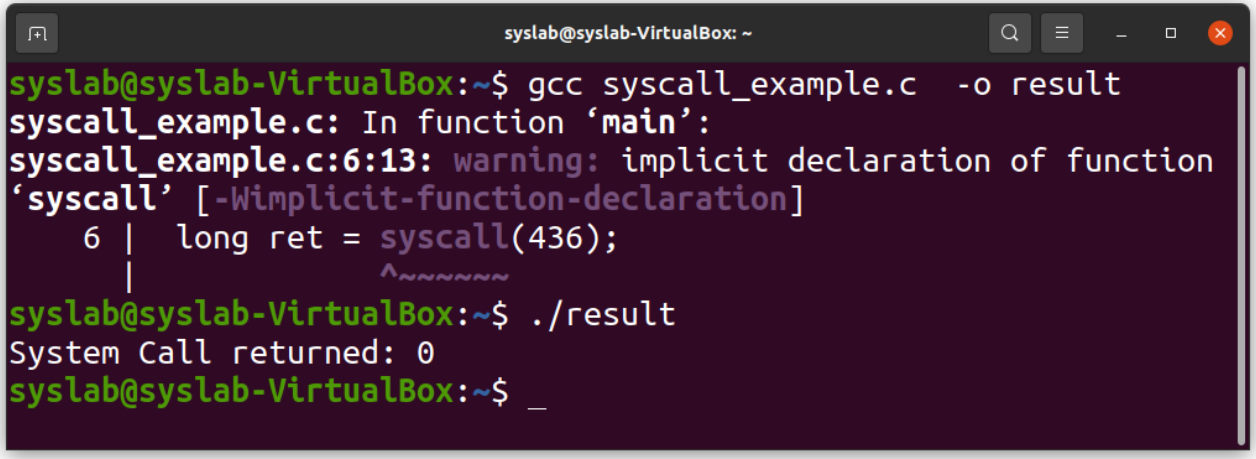
- sys_mycall usage example
  - $ vim syscall_example.c
    - For the x86 version, the syscall number is 436
      For the arm64 version, the syscall number is 460

```c
#include <stdio.h>
#include <sys/syscall.h>

int main(void)
{
        long ret = syscall(436);
        printf("System Call returned: %ld\n", ret);

        return 0;
}
```

  - $ gcc syscall_example.c -o result

System call usage

- my_syscall usage example
  - $ ./result



  - $ sudo dmesg

```
[    7.019395] *** VALIDATE vboxsf ***
[    7.019399] vboxsf: Successfully loaded version 6.1.38 r153438
[    7.019443] vboxsf: Successfully loaded version 6.1.38 r153438 on 5.4.214syslab SMP
[    7.021038] 13:07:04.467452 automount vbsvcAutomounterMountIt: Successfully mounted
[   12.393764] rfkill: input handler disabled
[  143.796105] hrtimer: interrupt took 10988967 ns
[  166.464380] System Call Example!
```