

# Project 1 Report

张天启 515030910612

October 6, 2017

采用kernel版本: 4.13.4  
OS版本: Ubuntu 16.04.2

## 1 编译内核

切换至root

```
sudo su
```

把下载好的内核文件放到/usr/src目录下，进入这个目录解压

```
xz -d linux-4.13.4.tar.xz  
tar -xvf linux-4.13.4.tar
```

上面解压失败的话先试试装这个包

```
apt-get install libncurses5-dev
```

NOTE: 编译失败或者不是第一次编译的时候最好先清空一下编译生成文件

```
make mrproper
```

或者

```
make clean
```

配置内核选项

```
make menuconfig
```

可能会提示少包，或者

```
make oldconfig
```

编译内核之前先检查有没有这个包

```
apt-get install libssl-dev
```

编译, n为线程数

```
make -jn  
make modules_install  
make install  
reboot
```

或者这样编译

```
make bzImage  
make modules  
make modules_install  
make install  
mkinitramfs 4.13.4 -o /boot/initrd-4.13.4.img  
update-grub2  
reboot
```

重启后查看版本

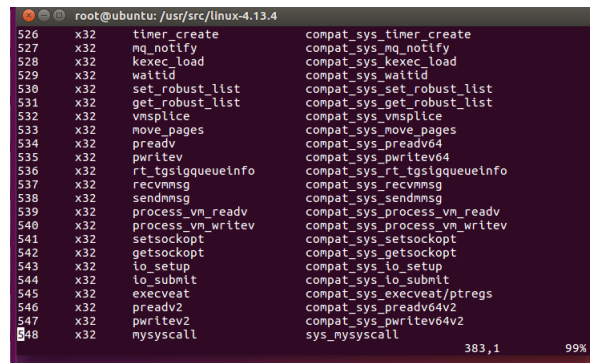
```
uname -a
```

## 2 添加系统调用

先查看系统的调用号使用了多少, x64或者x32都可以(如果你是32位机x32就行),在最后添加自己的系统调用名称和函数名称

```
vim /usr/src/linux-4.4.25/arch/x86/entry/syscalls/syscall_64.tbl
```

Figure 1: 查看系统调用编号

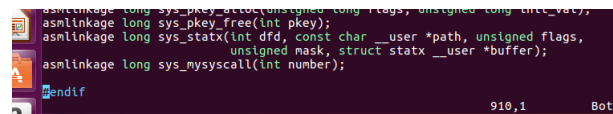


526	x32	timer_create	compat_sys_timer_create
527	x32	mq_notify	compat_sys_mq_notify
528	x32	kexec_load	compat_sys_kexec_load
529	x32	waitid	compat_sys_waitid
530	x32	set_robust_list	compat_sys_set_robust_list
531	x32	get_robust_list	compat_sys_get_robust_list
532	x32	vmsplice	compat_sys_vmsplice
533	x32	move_pages	compat_sys_move_pages
534	x32	preadv	compat_sys_preadv64
535	x32	pwritev	compat_sys_pwritev64
536	x32	rt_tgsigqueueinfo	compat_sys_rt_tgsigqueueinfo
537	x32	recvmsg	compat_sys_recvmsg
538	x32	sendmsg	compat_sys_sendmsg
539	x32	process_vm_readv	compat_sys_process_vm_readv
540	x32	process_vm_writev	compat_sys_process_vm_writev
541	x32	setsockopt	compat_sys_setsockopt
542	x32	getsockopt	compat_sys_getsockopt
543	x32	io_setup	compat_sys_io_setup
544	x32	io_submit	compat_sys_io_submit
545	x32	execveat	compat_sys_execveat/ptregs
546	x32	preadv2	compat_sys_preadv64v2
547	x32	pwritev2	compat_sys_pwritev64v2
548	x32	mysyscall	sys_mysyscall
			383,1 99%

---

## 声明系统调用函数

Figure 2: 声明系统调用



```
asmlinkage long sys_pkey_free(unsigned long flags, unsigned long ttt_val);
asmlinkage long sys_pkey_free(int pkey);
asmlinkage long sys_statx(int dfd, const char __user *path, unsigned flags,
                          unsigned mask, struct statx __user *buffer);
asmlinkage long sys_mysyscall(int number);

#endif
```

910,1 Bot

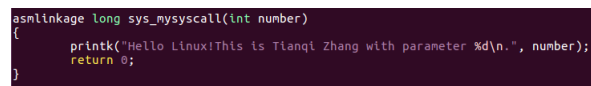
```
vim include/linux/syscalls.h
```

---

## 实现系统调用函数

```
vim kernel/sys.c
```

Figure 3: 实现系统调用函数



```
asmlinkage long sys_mysyscall(int number)
{
    printk("Hello Linux! This is Tianqi Zhang with parameter %d\n.", number);
    return 0;
}
```

---

## 重新编译内核

---

## 编写测试代码

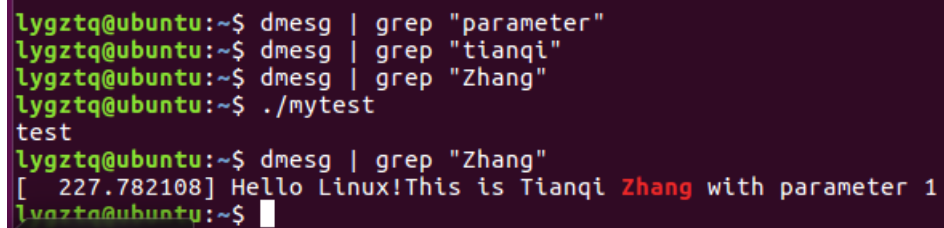
```
#include<unistd.h>
#include<stdio.h>
```

```
int main()
{
    printf("test.\n");
    syscall(548,1);
    return 0;
}
```

---

因为我用的是printk，所以要用dmesg查看

Figure 4: 查看结果



```
lygztq@ubuntu:~$ dmesg | grep "parameter"
lygztq@ubuntu:~$ dmesg | grep "tianqi"
lygztq@ubuntu:~$ dmesg | grep "Zhang"
lygztq@ubuntu:~$ ./mytest
test
lygztq@ubuntu:~$ dmesg | grep "Zhang"
[ 227.782108] Hello Linux!This is Tianqi Zhang with parameter 1
lygztq@ubuntu:~$
```