

西安电子科技大学
计算机学院

实
验
报
告

题目： Linux 系统调用的添加

班级： 1403013

姓名： 刘欢

学号： 14030130097

一、理论分析

系统调用是操作系统内核提供的有效服务界面，为了和用户空间上运行的进程进行交互的一组接口，通过该接口，应用程序可以访问硬件设备和其他操作系统资源。

在 x86 兼容机上，早起的 linux 内核使用 INT \$0x80 软中断来进行系统调用处理。用户进行系统调用时，将系统调用编号及其他参数压入寄存器中，然后触发中断，进入中断处理函数。中断处理函数将 CPU 从用户态切换到内核态，通过中断编号从系统调用表中找到对应的系统调用函数的地址，处理后获取系统调用的返回值，最后切换回进行调用的应用程序中。

二、设计与实现

经过查找资料和分析后得知，实现系统调用的方法之一是通过重新编译内核来添加系统调用，实现方法如下：

在 arch/x86/syscalls/syscall_64.tbl 表中新增一行，指定系统调用的编号，调用函数入口，并实现相应的函数，然后重新编译操作系统内核，最后加载新的操作系统内核就可以实现系统调用。具体步骤如下：

修改 arch/x86/syscalls/syscall_64.tbl 文件，在文件末尾添加代码：common my_syscall sys_my_syscall 并给定合适的调用编号。

实现调用函数 sys_my_syscall:

```
asmlinkage int sys_my_syscall(int number) {
    printk("This is my first syscall!");
    return number;
}
```

该函数向系统内核日志输出一段字符 “This is my first syscall!” 并返回参数 number。

打开 /usr/src/linux-source-2.6.38/arch/x86/kernel/syscall_table_32.S

添加一行，并记住其位置（编号）。

.long sys_mycall

打开 /usr/src/linux-source-2.6.38/arch/x86/include/asm/unistd_32.h

在合适的位置（根据上面的编号）添加一行：

```
#define __NR_mycall N /*N 为上面的编号*/
```

编译内核，步骤如下：

进入源代码目录：

1) #cd /usr/src/linux-source-3.2.0

linux-source-3.2.0 为源码目录名，可根据情况替换。

2) 清理以前编译留下的临时文件，

```
#make mrproper
```

3) 使用 make localmodconfig 自动精简内核

```
#make localmodconfig
```

如不知该如何精简可一路回车默认。

4) 开始编译

```
#make-kpkg clean
```

```
#fakeroot make-kpkg --initrd --append-to-version=-xidian kernel_image
```

安装内核

编译好的内核在上一层目录，执行命令：

```
#cd ..
```

```
#dpkg -i linux-image-3.2.39-xidian_3.2.39-xidian-10.00.Custom_i386.deb
```

linux-image-3.2.39-xidian_3.2.39-xidian-10.00.Custom_i386.deb 是编译后的安装文件名，可根据情况替

换。

重启验证内核

#reboot

重启后打开终端输入

#uname -r

查看版本号，如果是你定义的那个版本号就说明内核替换成功。如果发现已经安装的新内核包出现问题不合适，可以在启动 Grub 中选择原来的内核进入系统，然后删除新内核推到来过。

#dpkg -P linux-image-3.2.39-xidian_3.2.39-xidian

linux-image-3.2.39-xidian_3.2.39-xidian

测试：

编写程序：

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

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

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

```
int main()
```

```
{
```

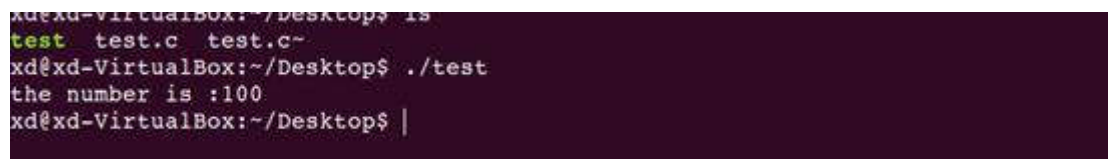
```
printf("the number is:  %d\n",syscall(223,100));
```

```
}
```

（223为调用函数编号）

编译并执行，应输出： the number is : 100

三、实验结果



测试程序输出一个字符串并调用系统调用函数返回给定参数 100，根据实验结果，系统调用成功，返回参数数值与预期结果一致。

四、心得与收获

通过本次实验，我从代码层面认识了一遍Linux系统实现系统调用的具体过程，对操作系统课程中所讲的理论知识有了直观的理解。在此过程中，我掌握了Ubuntu系统中实现系统调用的具体步骤，认识到在修改内核代码，编译、安装内核中需要注意的一些问题，有了巨大的收获。