**操作系统课程设计实验报告**

实验题目：实验一 Linux内核编译及添加系统调用

姓 名：张孜远

学 号：20151521

组 号：04

专 业：卓越学院 智能计算与数据科学

班 级：20186211

老师姓名：任彧老师

日 期：2022年12月20日

目 录

[一 题目介绍 1](#_Toc122502558)

[二 实验内容与思路 1](#_Toc122502559)

[三 遇到问题及解决方法 1](#_Toc122502560)

[四 核心代码及实验结果展示 1](#_Toc122502561)

[五 个人实验改进与总结 6](#_Toc122502562)

[5.1 个人实验小结 6](#_Toc122502563)

[5.2 个人实验总结 7](#_Toc122502564)

# 一 题目介绍

**实验目的：**

Linux 是开源操作系统，用户可以根据自身系统需要裁剪、修改内核，定制出功能更加合适、运行效率更高的系统，因此编译 Linux 内核是进行内核开发的必要基本功。在系统中根据需要添加新的系统调用是修改内核的一种常用手段，通过本次实验，我应理解 Linux 系统处理系统调用的流程以及增加系统调用的方法。

# 二 实验内容与思路

**实验内容：**

1、内核修改时有自己标签，并用dmesg验证；

2、Linux内核标签（系统启动显示一次）；

3、显示当前系统名称和版本的系统调用（内核、用户都有显示）；

4、修改nice和prio值的系统调用功能（内核、用户都有显示）；

5、改变主机名称为自定义字符串的系统调用（内核、用户都有显示）；

# 三 遇到问题及解决方法

**遇到的问题：**

1、编译内核的速度过慢：

在虚拟机软件VMware Workstation Pro中为子系统分配更多的硬件资源，其实本质是分配更多的CPU内核数，所以可以在编译时通过添加-j参数并行编译，例如：-j6就代表6核CPU内核数，-j8就代表8核CPU内核数。（后来了解到：这里对几核CPU的设置与一开始设置的虚拟机内核数量多少无关，直接-j8拉满即可，只要电脑不崩溃，都是可行的。）

2、编译内核时报错：

1）查看了Linux内核源代码后，了解到：SYSCALL\_DEFINE这一系列宏函数，在接受参数时需要将“参数类型”与“参数名称”分开来传递。、

2）由于内核版本、自己编写的C语言程序等诸多原因，通过网上查阅资料后得知需要在源码中更改部分设置。

**实验方法：**

从清华镜像网站中下载Linux源代码并对其进行编辑，编辑完自己的内核后，重启Linux虚拟机选择自己编辑的内核进行启动，使用相关指令测试相关功能。

# 四 核心代码及实验结果展示

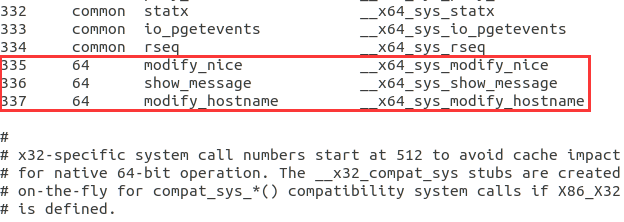
**实验过程和结果：**

0、首先创建虚拟机，分配60G内存与8核CPU；将从清华镜像网站中下载好的Linux内核源代码放入usr/src/文件夹中；

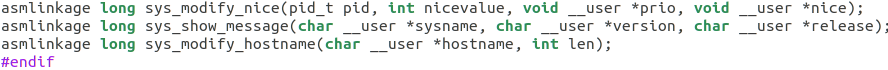
1、分配系统调用号，修改系统调用表：

查看系统调用表（arch/x86/entry/syscalls/syscalls\_64.tbl），可得：每个系统调用在表中占一个表项，其格式为：<系统调用号><commom/64/x32><系统调用名><服务例程入口地址>，其中第二项也称为应用二进制接口。

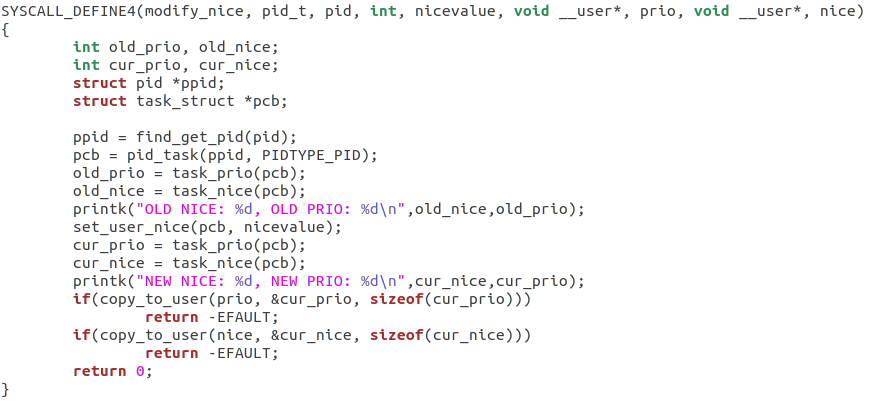
应用二进制接口分为三种：64、x32和common，即三种不同的调用约定。

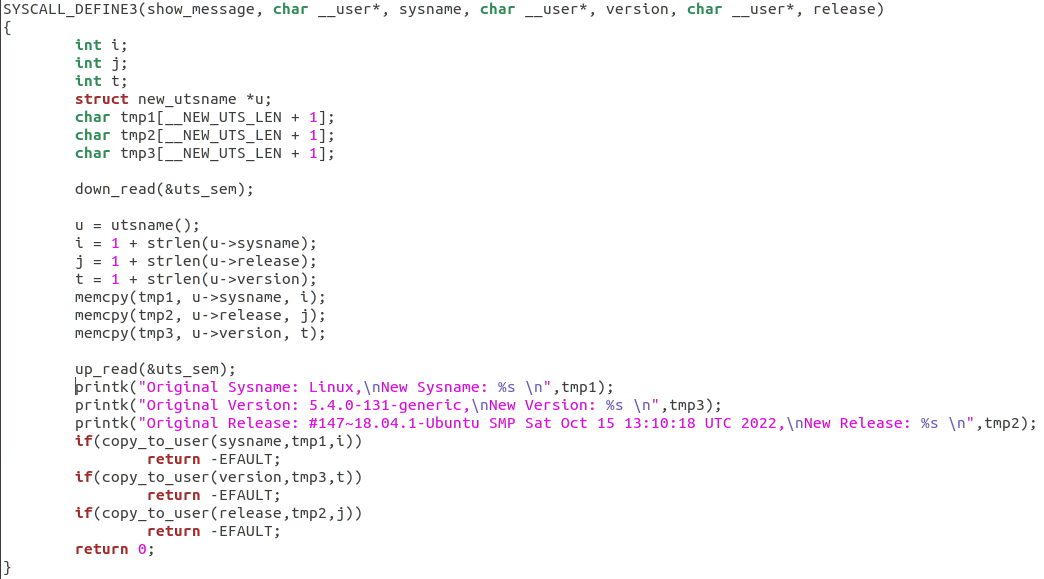
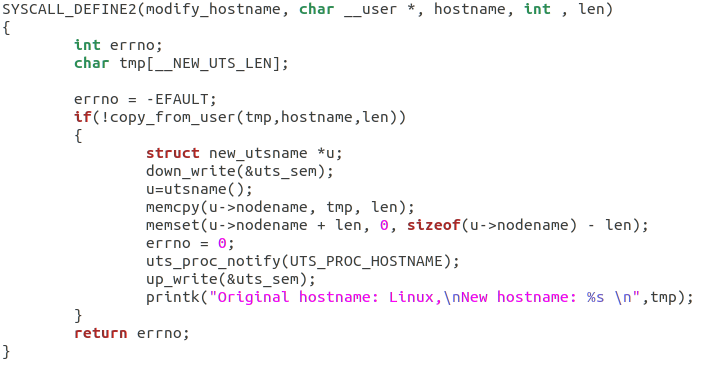
选择一个未使用的系统调用号进行分配，则新添加的系统调用号可使用335号。确定调用号后，应在系统调用表中关联新调用的调用号与服务例程入口，即在syscall\_64.tbl 文件中为新调用添加一条记录，修改结果如下图所示：

2、申明系统调用服务例程原型：

Linux系统调用服务例程的原型声明在文件Linux-4.12/include/linux/syscalls.h中，可在文件末尾添加下图内容：

3、实现系统调用服务例程：

 下面为新调用的modify\_nice、show\_message和modify\_hostname编写服务例程，通常添加在linux-4.19.25/kernel/sys.c文件中，如下图所示：



函数说明：

第三步与第二步的关系，类比为C语言中头文件与函数实现的关系，即：第二步对函数进行了声明，第三步给函数一个具体的实现。

以modify\_nice为例：该系统调用需要具备对指定进程的nice值的修改及读取的功能，同时返回进程最新的nice值及优先级prio，实现步骤分解如下：

1）根据进程号pid找到相应的进程控制块进程控制块PCB（因为进程控制块PCB中记录了用于描述进程情况及控制进程运行所需要的全部信息）；

2）根据进程控制块PCB读取它的nice值和优先级prio；

3）根据进程控制块PCB修改相应进程的nice值；

4）将得到的nice值和优先级prio进行返回。

4、重新编译内核：

上面三个步骤已经完成添加一个新系统调用的所有工作，但是要让这个系统调用真正在内核中运行起来，还需要重新编译内核。

Linux内核编译步骤：

1. 下载实验环境Linux虚拟机：VMware Workstation Pro；
2. 下载内核源码：来自清华镜像；
3. 解压缩内核源码文件；
4. 使用make mrproper指令清除残留的.config和.o文件；
5. 使用make menuconfig指令配置内核；

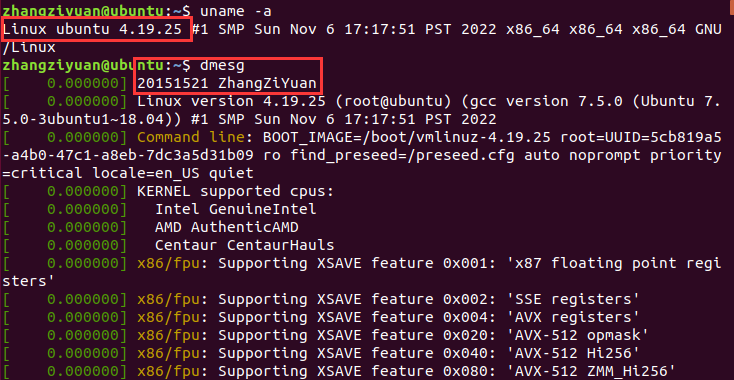
执行make menuconfig指令配置选项，均采取默认的模式，生成.config配置文件；

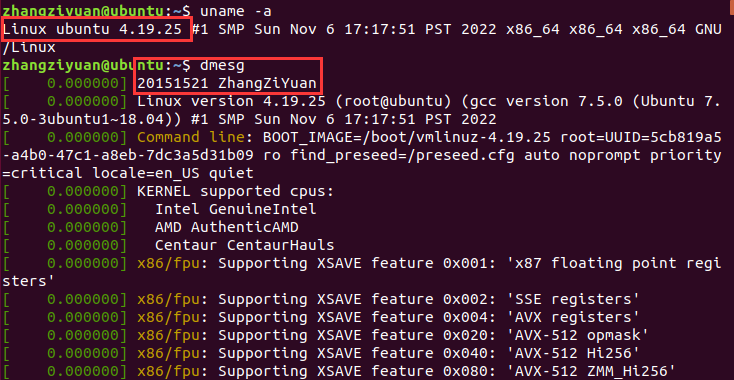
1. 使用make -j8指令编译内核（速度更快），生成启动映像文件；
2. 使用make modules指令编译模块；
3. 使用make modules\_install指令和make install指令安装内核；

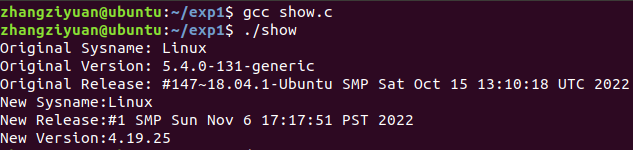
10、使用update-grub2指令配置grub引导程序；

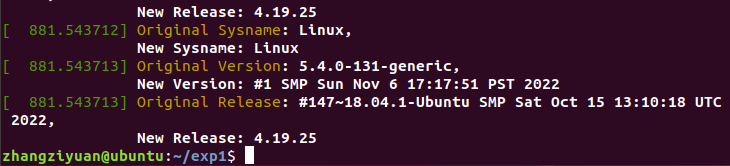
11、使用reboot指令重启系统，选择新编译的内核；

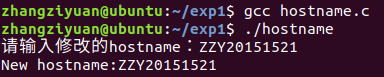
实验结果如下图所示：

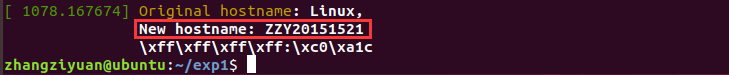
 1、重新启动，选择编译过的内核版本，查看新的系统版本：

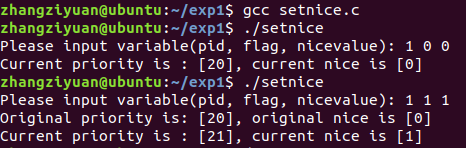
2、使用dmesg指令查看启动标签：（写入内核中）

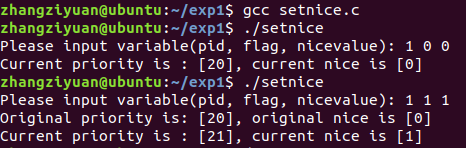
3.1、查看系统版本号信息：

3.2、使用dmesg指令查看内核中输出内容：

4.1、修改Linux中hostname名称：

4.2、使用dmesg指令查看内核中输出内容：

5.1、查看进程pid=1的nice和prio值：

5.2、修改并查看进程pid=1的nice和prio值：

5.3、使用dmesg指令查看内核中输出内容：

# 五 个人实验改进与总结

## 5.1 个人实验小结

|  |  |
| --- | --- |
| 项目实现创新点说明 | 1. 查看Linux内核源代码，了解Linux下的文件目录结构，以及根目录下的每个目录一般会存放什么形式、什么功能的文件：   /：根目录；  bin：一般用户可用，启动时会用到的命令，即在文件系统还没有被挂载时，也能够使用的命令；  boot：1、grub：开机设置相关文件；  2、内核文件（vmlinuz）  dev：存放设备文件  etc：1、rc.d：用于存放不同运行等级的启动脚本的链接文件；  2、X11  etc文件夹包含系统特有的可编辑配置文件，即用于控制程序运行的本地文件；  home：用户目录；  lib：存放程序的动态库和模块文件；  media：挂载本地磁盘或其他存储设备；  mnt：挂载其他临时文件系统；  opt：发行版附加的一些软件包的安装目录；  root：root用户的家；  sbin：存放很多只有root用户才能执行的命令，以及一些系统的更新、备份、还原和开关机用到的命令；  srv：存放服务进程所需的数据文件（如www网络服务器和ftp服务）和一些服务的执行脚本；  tmp：存放各种临时文件；  usr：包含诸多文件内容：  bin文件夹中存放非必要可执行文件（在单用户模式中不需要），面向所有用户；  include文件夹中存放了标准头文件；  lib文件夹中存放了/usr/bin/和/usr/sbin/中二进制文件；  local文件夹中存放了本地数据的第三层次，具体到本台主机；  share文件夹中存放了体系结构无关(共享)数据。  sbin文件夹中存放了非必要的系统二进制文件。例如：大量网络服务的守护进程；  src文件夹中存放了源代码，例如：内核源代码及其头文件。不过一般的发行版是不会保留内核源码，需要用户自己下载安装；  整个usr文件夹用于存储只读用户数据的第二层次；包含绝大多数的(多)用户工具和应用程序；  var：变量文件，在正常运行的系统中其内容不断变化的文件，如日志、脱机文件和临时电子邮件文件；有时是一个单独的分区； |

## 5.2 个人实验总结

**实验体会：**

这是操作系统的第一个实验，对于从来没有接触过Linux系统的我来说，颇具挑战性。首先，在编译内核的时候，需要先对Linux内核源码有了一定的了解（即代码的逻辑结构），我先花了3天时间阅读核心的Linux内核源码；同时在添加系统调用的时候，更需要对内核中的各个源函数的功能通过查阅资料进行了解，达到能够自主修改的程度。

在做完第一个操作系统实验之后，有种苦难过后的重生感。一方面是感叹一开始接触新知识就要掌握并应用它确实很难，但在刻苦努力之后的成功，特别是自己添加的系统调用函数能够显示正确结果，这更具成就感。

**六 参考文献**

[1] kernel环境配置.veristas501. <https://veritas501.space/2018/06/03/kernel%E7%8E%AF%E5%A2%83%E9%85%8D%E7%BD%AE/>

[2] set\_user\_nice源码. <https://code.woboq.org/linux/linux/kernel/sched/core.c.html#set_user_nice>

[3] Completely Fair Scheduler.wikipedia. <https://en.wikipedia.org/wiki/Completely_Fair_Scheduler>

[4] CFS Scheduler.The Linux Kernel documentation. <https://www.kernel.org/doc/html/latest/scheduler/sched-design-CFS.html>

[5] Linux内核源码

https://elixir.bootlin.com/linux/v5.10.109/source

[6] Linux内核编译错误：make[1]:“debian/canonical certs.pem” https://blog.csdn.net/m0\_51203305/article/details/120805372

[7] sudo make modules\_install报错

https://blog.csdn.net/waterwhoami/article/details/110621237

[8] 如何在Ubuntu20.04编译内核边添加一个系统调用

https://blog.csdn.net/bar\_workshop/article/details/111647568