实验三 Proc文件系统

proc 文件系统是一种内核和内核模块用来向进程 (process) 发送信息的机制 (所以叫做 /proc)。这个伪文件系统让你可以和内核内部数据结构进行交互，获取有关进程的有用信息。

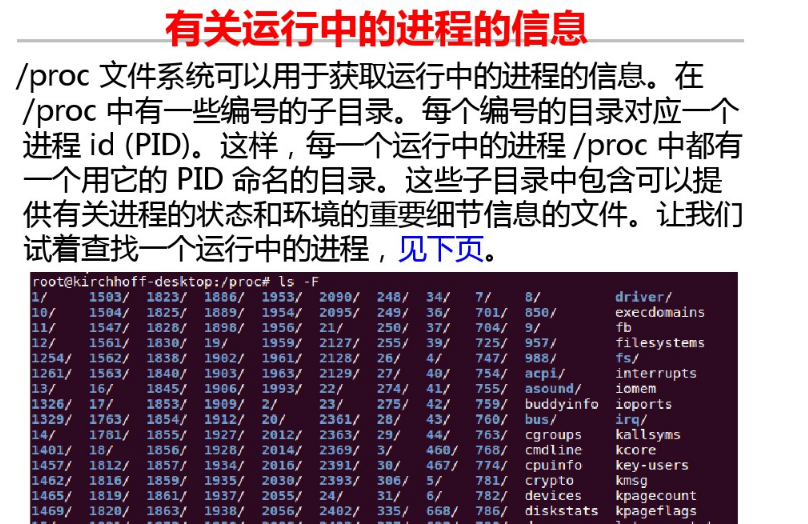
（1）实验目的：掌握proc文件系统的工作原理

（2）实验内容：编写文件名proctest的proc文件，实用cat命令读这个文件，输出的结果为helloworld!

实验内容



1. Proc文件系统解释  
   /proc文件系统是一个虚拟的文件系统，在该目录中所有的文件都不会占用磁盘空间，这些数据都是linux内核在启动后动态生成并保存在内存中的。





/proc/系统中每个进程的目录都包含了同样的项目。

cmdline:该目录包含了启动该进程的整个命令行。

cwd:符号连接指向进程的当前工作目录。

environ:该文件包含了为这个进程定义的所有的环境变量，格式为“变量=值”

exe：这是指向该进程所执行的可执行程序文件的符号链接。

fd:此目录包含了该进程打开的所有描述符。

maps：当大因该文件内容时，将看到该进程的地址空间中当前映射到文件中的那部分，自左至右各列为与该映射关联的地址空间，与该映射关联的权限，该映射起始位置的偏移量，保存映射文件设备的主，从序列（16进制），inode序号，以该文件的文件名。

root：该符号连接指向该进程使用的根目录。

status：该文件包含进程的众多信息，即可执行文件名，当前状态，PID和PPID，实际及有效的UID和GID，内存使用装态。

/proc还包含了机器上的各种硬件信息。

cat /proc/devices //查看系统中的设备信息，包括字符设备和块设备

cat /proc/cpuinfo //查看cpu信息

cat /proc/filesystems //查看系统支持的文件系统

cat /proc/ioports //查看系统的i/o端口

cat /proc/interrupts //查看中断信息

cat /proc/modules //查看系统内核载入的模块列表

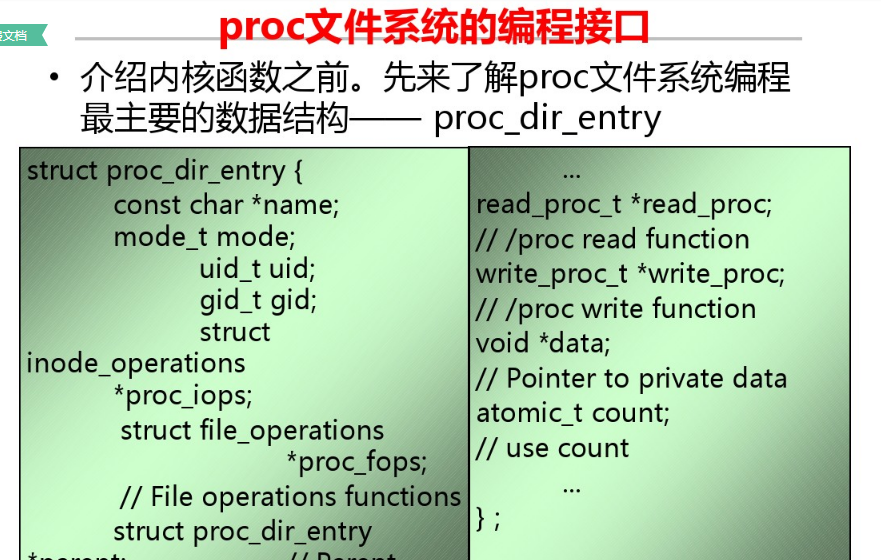
cat /proc/mounts //查看挂载的文件系统

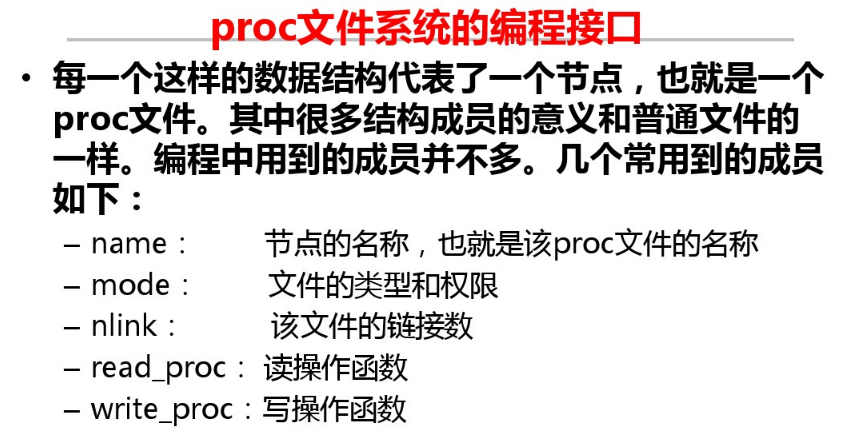
cat /proc/swaps //查看交换分区的使用情况

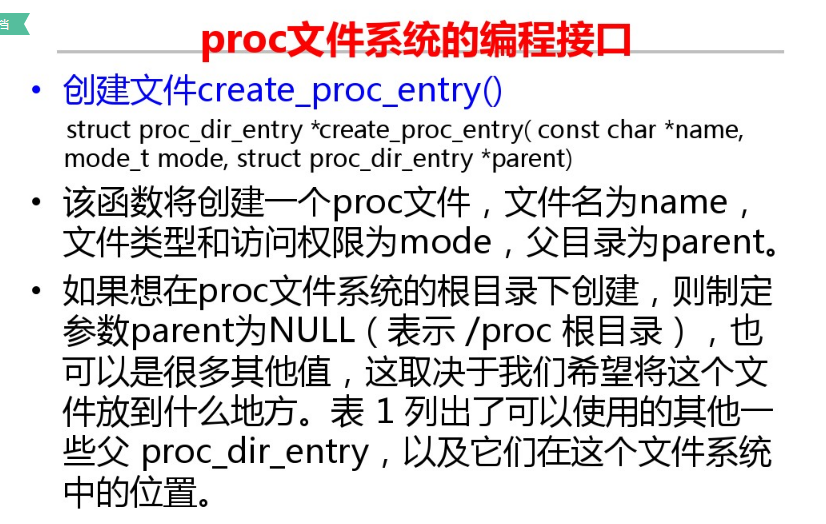
cat /proc/partitions //查看系统分区信息

cat /proc/version //查看当前内核的版本

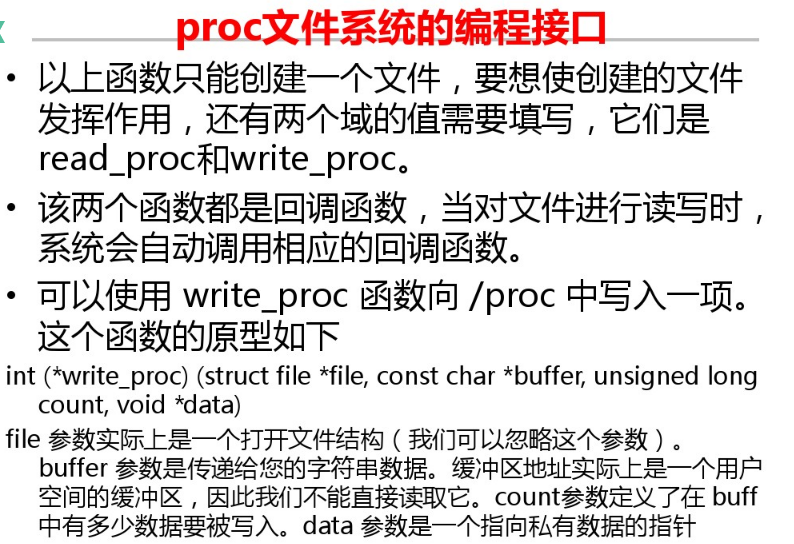
（2）

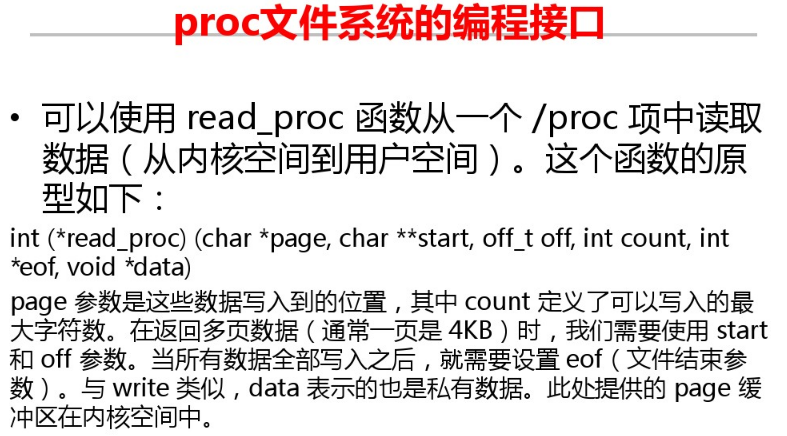












hello.c

#define \_\_KERNEL\_\_

#define MODULE

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/init.h>

#include <linux/proc\_fs.h> /\* Necessary because we use the proc fs \*/

#define procfs\_name "proctest"

struct proc\_dir\_entry \*Our\_Proc\_File;

int procfile\_read(char \*buffer,

char \*\*buffer\_location,

off\_t offset, int buffer\_length, int \*eof, void \*data)

{

int ret;

ret = sprintf(buffer, "HelloWorld!\n");

return ret;

}

int proc\_init()

{

Our\_Proc\_File = create\_proc\_entry(procfs\_name, 0644, NULL);

if (Our\_Proc\_File == NULL) {

remove\_proc\_entry(procfs\_name, NULL);

printk(KERN\_ALERT "Error: Could not initialize /proc/%s\n",

procfs\_name);

return -ENOMEM;

}

Our\_Proc\_File->read\_proc = procfile\_read;

Our\_Proc\_File->owner = THIS\_MODULE;

Our\_Proc\_File->mode = S\_IFREG | S\_IRUGO;

Our\_Proc\_File->uid = 0;

Our\_Proc\_File->gid = 0;

Our\_Proc\_File->size = 37;

printk("/proc/%s created\n", procfs\_name);

return 0;

}

void proc\_exit()

{

remove\_proc\_entry(procfs\_name, NULL);

printk(KERN\_INFO "/proc/%s removed\n", procfs\_name);

}

module\_init(proc\_init);

module\_exit(proc\_exit);

Makefile

ifneq ($(KERNELRELEASE),)

obj-m := hello.o

else

KDIR := /lib/modules/2.6.29/build

all:

make -C $(KDIR) M=$(PWD) modules

clean:

rm -f \*.ko \*.o \*.mod.o \*.mod.c \*.symvers

endif

加载模块，在proc目录下看是否有proctest proc文件。

在proc目录下执行

cat proctest