## 实验八：设备管理观察

### 实验目的：

1. 掌握设备管理的基本命令
2. 可以获取和设置系统当前设备相关的主要信息

### 预备知识：

1. 基本命令

|  |  |
| --- | --- |
| 命令名 | 功能说明 |
| mknod | 建立块/字符特殊文件 |
| dislocate | 使进程和终端断开连接或重新连接 |
| getty | 设置终端工作方式 |
| stty | 改变/查询终端行设置 |
| setterm | 设置终端属性 |
| tset | 终端初始化 |
| tput | 初始化终端或查询terminfo数据库 |
| resizecons | 改变控制台尺寸的核心数据 |
| unicode\_start | 使控制台在Unicode方式下工作 |
| unicode\_stop | 使控制台不在Unicode方式下工作 |
| kbd\_mode | 报告或设置键盘工作方式 |
| kbdrate | 重置键盘重复率和延迟时间 |
| loadkeys | 装入键盘转换表 |
| dumpkeys | 转储键盘转换表 |
| setmetamode | 定义键盘元键处理 |
| showkey | 检查键盘送来的扫描码和键码 |
| chvt | 改变前台虚拟终端 |
| fgconsole | 显示虚拟活动终端数 |
| deallocvt | 释放空闲的虚拟终端数 |
| openvt,open | 在一个新的虚拟终端上启动一个程序 |
| switchto | 切换至新的虚拟终端 |
| vlock | 锁住虚拟终端 |
| screen | VT100/ANSI终端仿真的屏幕管理器 |
| mev | 报告鼠标事件 |

1. /proc文件系统

|  |  |
| --- | --- |
| 文件(目录)名 | 内容说明 |
| /proc/devices | 主要的字符和块设备编号及分配给这些编号的驱动程序名字 |
| /proc/ioports | 各种设备驱动程序注册的I/O端口范围 |
| /proc/dma | 被驱动程序留作专用的DMA通道以及驱动程序赋予的名字 |
| /proc/scsi | scsi设备及其相关信息 |
| /proc/pci | PCI设备信息 |
| /proc/rtc | 硬件实时时钟的相关信息 |
| /proc/misc | 被内核函数misc\_register注册的驱动程序 |

1. 几类典型设备

|  |  |
| --- | --- |
| 设备文件名 | 设备(说明) |
| /dev/null | 用于不需存储的输出（虚拟字符设备） |
| /dev/zero | 用于二进制“0”的无限提供（虚拟字符设备） |
| /dev/random | 随机数池（虚拟字符设备） |
| /dev/urandom | 伪随机数池（虚拟字符设备） |
| /dev/ttyS0 | COM1 |
| /dev/ttyS1 | COM2 |
| /dev/lp0 | LPT1 |
| /dev/lp1 | LPT2 |
| /dev/psaux | PS/2端口 |
| /dev/fd0~/dev/fd7 | 软驱 |
| /dev/hda~/dev/hdh | IDE设备 |
| /dev/sda~/dev/sddx | SCSI设备 |

### 实验内容:

1. 利用手册页，学习设备相关主要命令（以上所列全部）的用法，并列出你的系统当前的信息。
2. mknod-建立块/字符特殊文件

该命令的标准形式为:

mknod DEVNAME {b | c} MAJOR MINOR

① DEVNAME是要创建的设备文件名，如果想将设备文件放在一个特定的文件夹下，就需要先用mkdir在dev目录下新建一个目录；

② b和c分别表示块设备和字符设备：

* b表示系统从块设备中读取数据的时候，直接从内存的buffer中读取数据，而不经过磁盘；
* c表示字符设备文件与设备传送数据的时候是以字符的形式传送，一次传送一个字符，比如打印机、终端都是以字符的形式传送数据；

③ MAJOR和MINOR分别表示主设备号和次设备号：

为了管理设备，系统为每个设备分配一个编号，一个设备号由主设备号和次设备号组成。主设备号标示某一种类的设备，次设备号用来区分同一类型的设备。Linux操作系统中为设备文件编号分配了32位无符号整数，其中前12位是主设备号，后20位为次设备号，所以在向系统申请设备文件时主设备号不要超过4095，次设备号不要超过220-1。

* 使用mknod命令创建主设备号为128，次设备号为512的字符特殊文件mydev1：

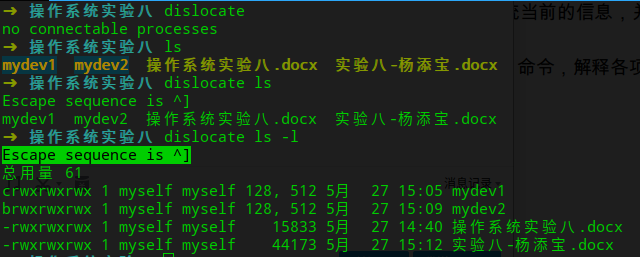


* 使用mknod命令创建主设备号为128，次设备号为512的块特殊文件mydev2：



1. dislocate-使进程和终端断开连接或重新连接

dislocate是一个简洁的工具，可以让人们通过伪终端把程序分离出来，然后，当需要这些程序的时候，在重新挂载它们。这个工具是专门为那些缓慢或不稳定的终端会话（它们很容易中断）而提供的。



1. getty-设置终端工作方式

是Unix类操作系统启动时必须的三个步骤之一，用来开启终端，进行终端的初始化，设置终端。

用法：

getty [-h][-d<组态配置文件>][-r<延迟秒数>][-t<超时秒数>][-w<等待字符串>][终端机编号][连线速率<终端机类型><管制线路>]

或

getty [-c<定义配置文件>]

getty命令设置和管理终端线路和端口。getty命令由init命令来运行。getty命令与终端状态管理员程序相链接。终端状态管理员程序提供了终端控制和登录的复合功能。 注意：getty命令不在命令行输入。

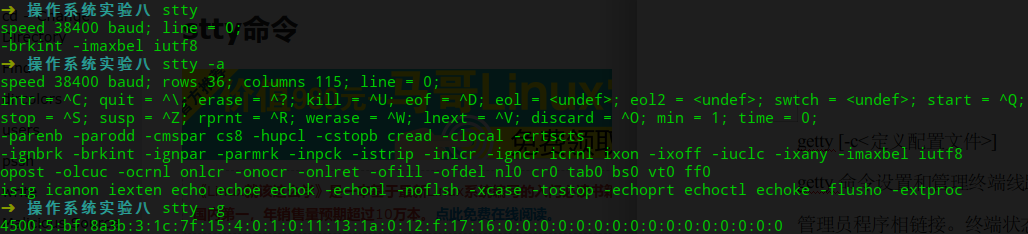
1. stty-改变/查询终端行设置

stty命令修改终端命令行的相关设置。

参数：

-a：以容易阅读的方式打印当前的所有配置；

-g：以stty可读方式打印当前的所有配置。



1. setterm-设置终端属性

setterm向终端写一个字符串到标准输出，调用终端的特定功能。在虚拟终端上使用，将会改变虚拟终端的输出特性。不支持的选项将被忽略。

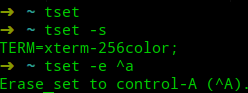
下图演示了打开粗体（额外亮度）模式，该命令只在ubuntu的终端下测试有效：



1. tset-终端初始化

# tset -s 查看终端类型

# tset -e ^? 键设置



1. tput-初始化终端或查询terminfo数据库

tput 用 System V 传统的 terminfo capability 为参数，它的数据库所在目录是：

/etc/terminfo ， /lib/terminfo ， /usr/share/terminfo/

举例：

# tput cup 10 30 光标位置

# tput civis 光标隐藏

# tput cnorm 光标重现

# tput smul 下划线

# tput rmul 取消下划线

# tput dl 10 删除下面十行

# tput bold 黑体

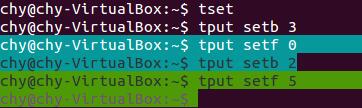
# tput rev 背景色与前景色反转

# tput cols 报告终端列数

# tput lines 报告终端行数

# tput sc 保存当前光标位置

# tput sgr0 恢复默认值



1. resizecons-改变控制台尺寸的核心数据

调用方法为：

resizecons COLSxROWS

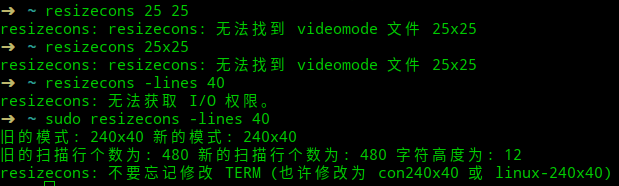
或

resizecons COLS ROWS

或

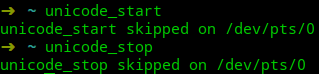
resizecons -lines ROWS

其中 ROWS 为 25, 28, 30, 34, 36, 40, 44, 50, 60 中的一个



1. unicode\_start-使控制台在Unicode方式下工作

unicode\_stop-使控制台不在Unicode方式下工作



1. kbd\_mode-报告或设置键盘工作方式

如果没有参数，kbd\_mode会显示当前键盘的模式，如果有参数，它会把键盘设置成相应的模式。

参数：

-s：键盘扫描码模式（原始）

-k：键值（keycode）模式（半原始）

-a : ASCII模式(XLATE)

-u : UTF-8模式(UNICODE)

XLATE模式是传统模式，所用的代码可以是任何8-bit(8位)的字符集。一般这个字符集同后面用到的字符集是匹配的，在它们被传给屏幕后，它们会根据consolechars -m选择的字符集在内部转换为Unicode。

在UNICODE模式，键盘会产生16位的字符，这些字符会以UTF-8编码形式传给内核。UTF-8在这后两种模式中要用到loadkeys定义的键盘映射表。

注意：如果不是把键盘模式改为ASCII或者Unicode，很可能会使键盘不可用。

这个命令也可以在有些程序使你的键盘处于错误状态时用来把键盘改回XLATE或者UNICODE模式（比如通过远程登录）。在有些过时的版本的程序中-u和-s是一样的。



1. kbdrate-重置键盘重复率和延迟时间

按键延时是只长按一个按键多少时间才会开始重复这个按键。开始重复过程后，字符会以一定频率出现(cps)，也就是重复频率。终端中，这些值可以通过 kbdrate 设置。

用法：

kbdrate [-d delay] [-r rate]

延迟200 ms重复频率是30 cps：



不加任何参数会还原到默认值250 ms和10.9 cps：



1. loadkeys-装入键盘转换表

loadkeys命令可以根据一个键盘定义表改变Linux键盘驱动程序转译键盘输入过程。

参数：

-v --verbose：印出详细的资料，你可以重复以增加详细度。

-q --quiet：不要显示任何讯息。

-c --clearcompose：清除所有 composite 定义。

-s --clearstrings：将定串定义表清除。

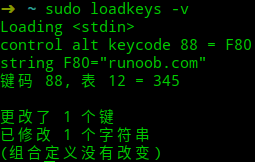
下面演示了定义按键组合，在执行完loadkeys命令后输入如下两行语句：

control alt keycode 88 = F80 //现确定键代码

string F80="runoob.com" //给变变量设定值

然后按下Ctrl+D键，确定输入。

执行后的效果是按下Ctrl+Alt+F12输出runoob.com



1. dumpkeys-转储键盘转换表

dumpkeys命令用于显示键盘映射表，输出的内容可以被loadkeys命令识别，改变映射关系。

参数：

-i 驱动信息(键码范围、数量、状态键)

-l 详细驱动信息

-n 十六进制显示

-f 显示全部信息

-1 分行显示按键组合

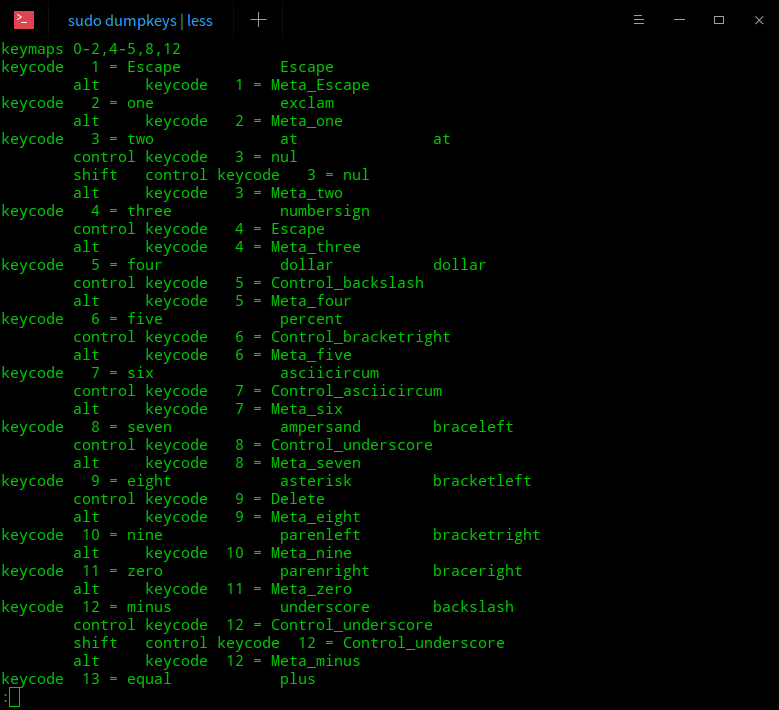
-S 设定输出格式(0：预设 1：完整 2：分行 3简单)

--funcs-only 功能键信息

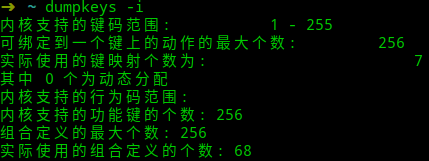
--keys-only 键组合信息

--compose-only 普通键信息

执行的dumpkeys | less后的部分结果如下：



显示驱动信息：



1. setmetamode-定义键盘元键处理

用法：

setmetamode [ meta|bit|metabit | esc|prefix|escprefix ]

没有参数时，setmetamode将打印当前Meta键模式；有参数时，设置所指出的Meta键模式。

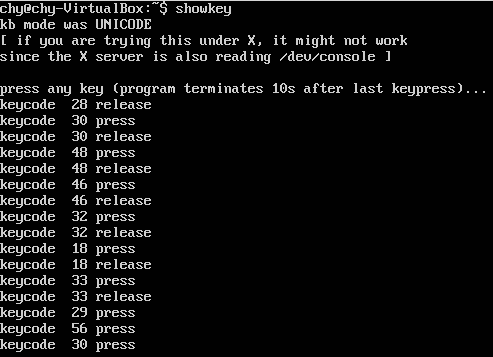
在图形界面下执行的结果：



在字符界面下执行的结果：



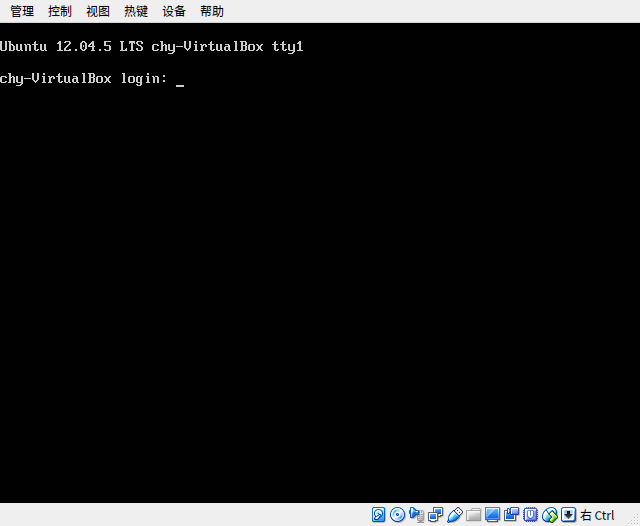
1. showkey-检查键盘送来的扫描码和键码



1. chvt-改变前台虚拟终端

chvt N命令用来生成/dev/ttyN的前台终端。如果它本来不存在，即创建相应的屏幕。为了删除掉不用的VT(虚拟终端)，可使用deallocvt。





1. fgconsole-显示虚拟活动终端数

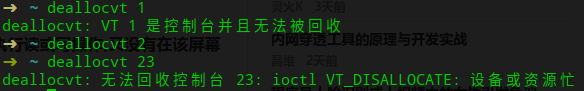


1. deallocvt-释放空闲的虚拟终端数

用法：

deallocvt [ N1 N2 ... ]

如果不指定参数，deallocvt程序会释放所有未使用的虚拟终端的核心内存和数据结构。如果给定了参数Ni那么就只释放TTY /dev/ttyNi.



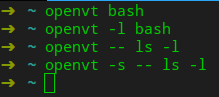
1. openvt, open-在一个新的虚拟终端上启动一个程序

在Debian系统内，open命令是openvt命令的符号链接，用于在一个新的虚拟终端中运行应用程序。

用法：

openvt [-c vtnumber] [-s] [-u] [-l] [-v] [--] command command\_options

openvt将会找到第一个可用的虚拟终端（VT），在上面以给定的参数command\_options运行给定的命令command，标准输入、标准输出和标准错误设备将被引导到那个终端，当前搜索路径（$PATH）被用来寻找请求的命令。如果没有找到命令将会使用环境变量$SHELL。





1. switchto-切换至新的虚拟终端

switchto将切换当前终端到指定终端。当系统热键不可用或在脚本中使用时该命令十分有用。

1. vlock-锁住虚拟终端

用法：

vlock [-achv]

参数：

-a或--all 　锁住所有的终端阶段作业，如果您在全屏幕的终端中使用本参数，则会将用键盘

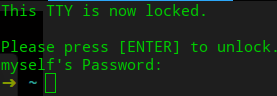
切换终端机的功能一并关闭。

-c或--current 　锁住目前的终端阶段作业，此为预设值。

-h或--help 　在线帮助。

-v或--version 　显示版本信息。





1. screen-VT100/ANSI终端仿真的屏幕管理器

GNU Screen是一款由GNU计划开发的用于命令行终端切换的自由软件。用户可以通过该软件同时连接多个本地或远程的命令行会话，并在其间自由切换。

GNU Screen可以看作是窗口管理器的命令行界面版本。它提供了统一的管理多个会话的界面和相应的功能。

* 会话恢复

只要Screen本身没有终止，在其内部运行的会话都可以恢复。这一点对于远程登录的用户特别有用——即使网络连接中断，用户也不会失去对已经打开的命令行会话的控制。只要再次登录到主机上执行screen -r就可以恢复会话的运行。同样在暂时离开的时候，也可以执行分离命令detach，在保证里面的程序正常运行的情况下让Screen挂起（切换到后台）。这一点和图形界面下的VNC很相似。

* 多窗口

在Screen环境下，所有的会话都独立的运行，并拥有各自的编号、输入、输出和窗口缓存。用户可以通过快捷键在不同的窗口下切换，并可以自由的重定向各个窗口的输入和输出。Screen实现了基本的文本操作，如复制粘贴等；还提供了类似滚动条的功能，可以查看窗口状况的历史记录。窗口还可以被分区和命名，还可以监视后台窗口的活动。

* 会话共享

Screen可以让一个或多个用户从不同终端多次登录一个会话，并共享会话的所有特性（比如可以看到完全相同的输出）。它同时提供了窗口访问权限的机制，可以对窗口进行密码保护。用法：

screen [-AmRvx -ls -wipe][-d <作业名称>][-h <行数>][-r <作业名称>][-s ][-S <作业名称>]

参数：

-A 将所有的视窗都调整为目前终端机的大小。

-d <作业名称> 将指定的screen作业离线。

-h <行数> 指定视窗的缓冲区行数。

-m 即使目前已在作业中的screen作业，仍强制建立新的screen作业。

-r <作业名称> 恢复离线的screen作业。

-R 先试图恢复离线的作业。若找不到离线的作业，即建立新的screen作业。

-s 指定建立新视窗时，所要执行的shell。

-S <作业名称> 指定screen作业的名称。

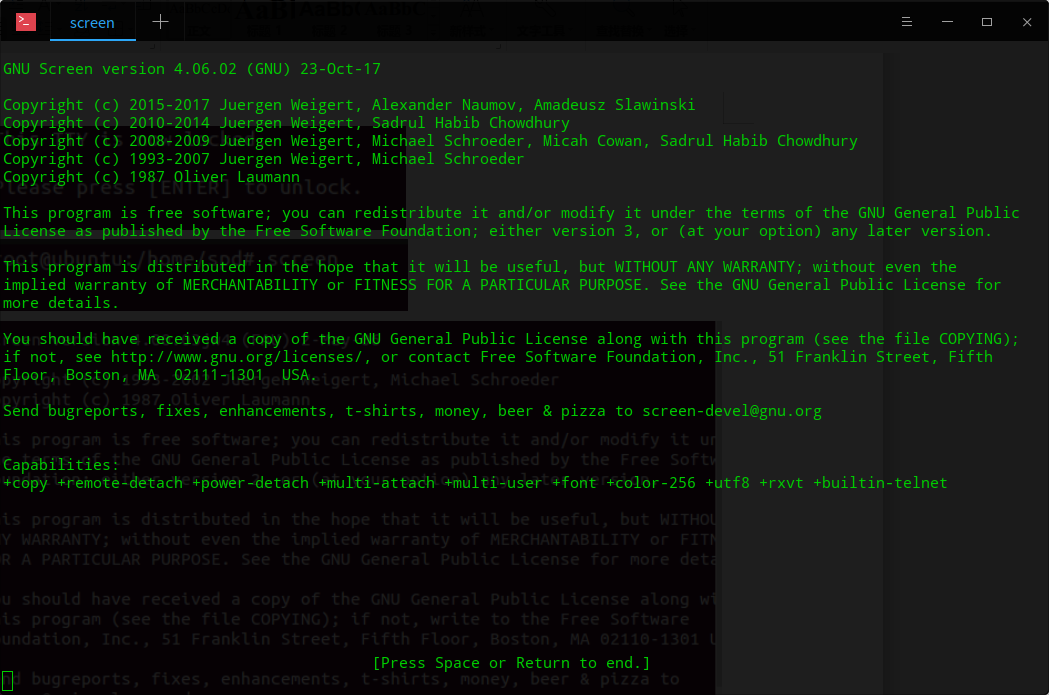
-v 显示版本信息。

-x 恢复之前离线的screen作业。

-ls或--list 显示目前所有的screen作业。

-wipe 检查目前所有的screen作业，并删除已经无法使用的screen作业。



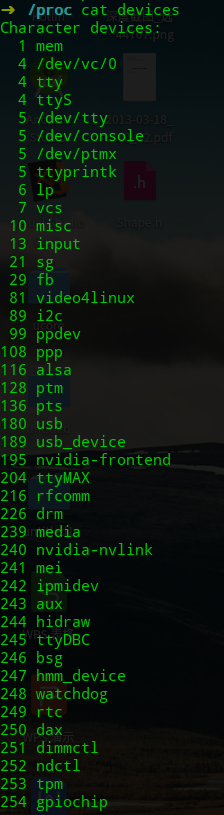
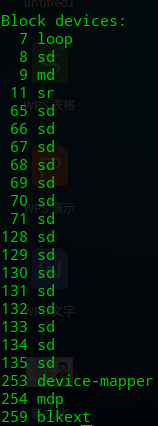


1. mev-报告鼠标事件

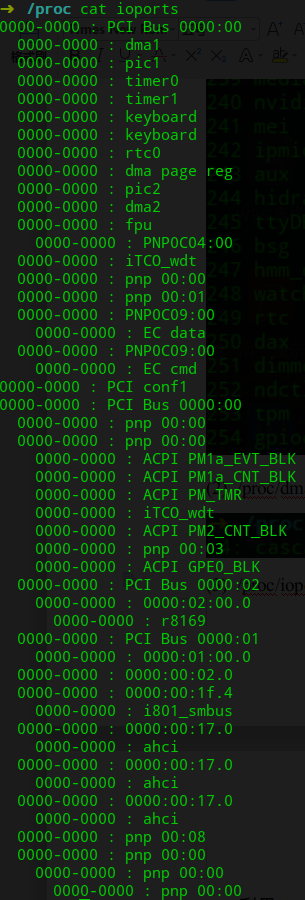
使用Ubuntu 12.04，输入mev命令后点击鼠标，将出现类似如下的结果：



1. 利用/proc文件系统，列出你的系统当前的信息，并解释相关内容。
2. /proc/devices-主要的字符和块设备编号及分配给这些编号的驱动程序名字

1. /proc/ioports-各种设备驱动程序注册的I/O端口范围



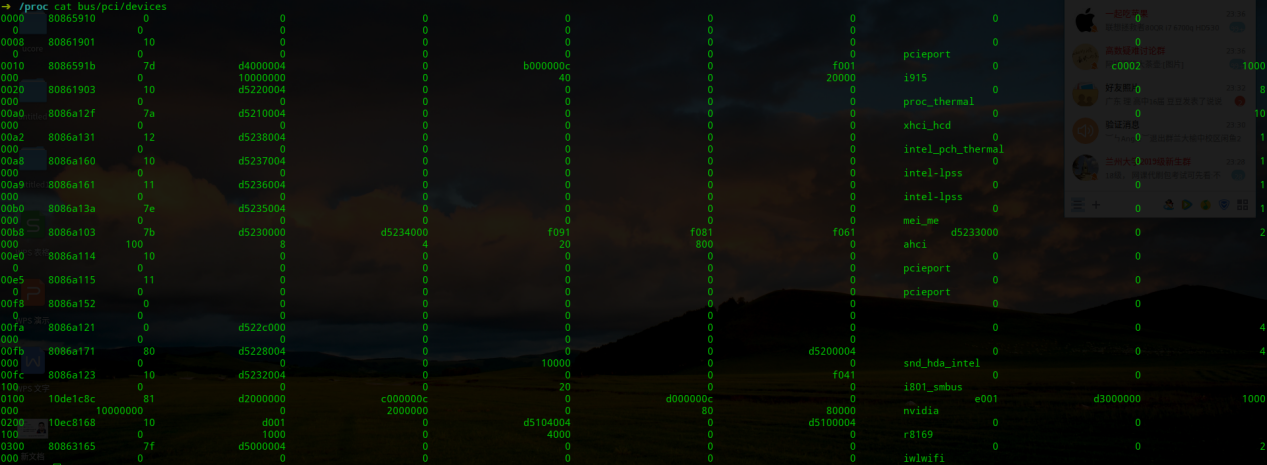
1. /proc/dma-被驱动程序留作专用的DMA通道以及驱动程序赋予的名字



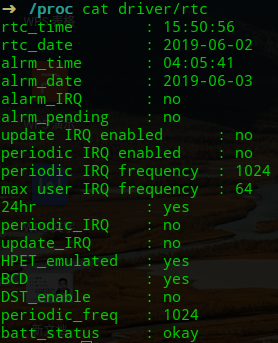
1. /proc/scsi-scsi设备及其相关信息



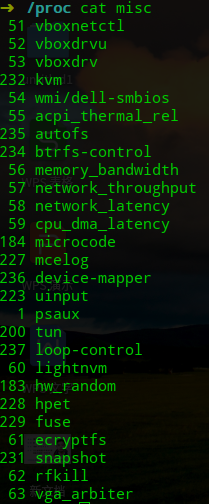
1. /proc/bus/pci-PCI设备信息



1. /proc/driver/rtc-硬件实时时钟的相关信息



1. /proc/misc-被内核函数misc\_register注册的驱动程序



1. 利用/proc文件系统，列出你的系统当前的信息，并解释相关内容观察/dev目录中的文件，使用ls –l命令，解释各项信息的含义。





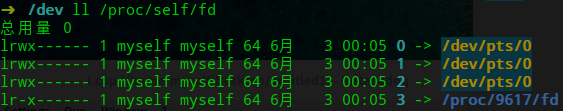












/dev/hd[a-t]：IDE设备

/dev/sd[a-z]：SCSI设备

/dev/fd[0-7]：标准软驱

/dev/md[0-31]：软raid设备

/dev/loop[0-7]：本地回环设备

/dev/ram[0-15]：内存

/dev/null：无限数据接收设备,相当于黑洞

/dev/zero：无限零资源

/dev/tty[0-63]：虚拟终端

/dev/ttyS[0-3]：串口

/dev/lp[0-3]：并口

/dev/console：控制台

/dev/fb[0-31]：framebuffer

/dev/cdrom => /dev/hdc

/dev/modem => /dev/ttyS[0-9]

/dev/pilot => /dev/ttyS[0-9]

/dev/random：随机数设备

/dev/urandom：随机数设备