

Tekkaman Ninja

tekkamanninja.blog.chinaunix.net

Linux我的梦想，我的未来！ 本博客的原创文章的内容会不定期更新或修正错误！转载文章都会注明出处，若有侵权，请即时同我联系，我一定马上删除！！原创文章版权所有！如需转载，请注明出处： tekkamanninja.blog.chinaunix.net ，谢谢合作！！ 拒绝一切广告性质的评论，一经发现立即举报并删除！

首页 | 博文目录 | 关于我



tekkamanninj

博客访问： 75909

博文数量： 263

博客积分： 15936

博客等级： 上将

技术积分： 13951

用户组： 普通用户

注册时间： 2007-03-27 11:22

加关注 短消息

论坛 加好友

个人简介

Fedora-ARM

文章分类

- 全部博文（263）
- Red Hat（2）

代码管理（6）

感悟（3）

Linux调试技术（2）

MaxWit（1）

Linux设备驱动程（41）

Android（20）

neo freerunner（2）

计算机硬件技术（9）

网络（WLAN or LA（8）

励志（7）

ARM汇编语言（1）

Linux操作系统的（15）

Linux内核研究（38）

ARM-Linux应用程（19）

建立根文件系统（4）

Linux内核移植（14）

Bootloader（45）

建立ARM-Linux交（7）

未分配的博文（19）

文章存档

2014年（1）

Linux设备驱动程序学习（4）-高级字符驱动程序操作 [（1）ioctl and llseek ]

2007-10-31 15:16:39

分类： LINUX

Linux设备驱动程序学习（4）

-高级字符驱动程序操作 [（1）ioctl and llseek]

今天进入《Linux设备驱动程序（第3版）》第六章高级字符驱动程序操作的学习。

一、ioctl

大部分设备除了读写能力，还可进行超出简单的数据传输之外的操作，所以设备驱动也必须具备进行各种硬件控制操作的能力。这些操作常常通过 ioctl 方法来支持，它有和用户空间版本不同的原型：

```
int (*ioctl) (struct inode *inode, struct file *filp,
              unsigned int cmd, unsigned long arg);
```

需要注意的是：不管可选的参数arg是否由用户给定为一个整数或一个指针，它都以一个unsigned long的形式传递。如果调用程序不传递arg参数，被驱动收到的 arg 值是未定义的。因为在arg参数上的类型检查被关闭了,所以若一个非法参数传递给 ioctl，编译器是无法报警的，且任何关联的错误难以查找。

选择ioctl命令

为了防止向错误的设备使用正确的命令，命令号应该在系统范围内唯一。为方便程序员创建唯一的 ioctl 命令代号，每个命令号被划分为多个位字段。要按 Linux 内核的约定方法为驱动选择 ioctl 的命令号，应该首先看看 include/asm/ioctl.h 和 Documentation/ioctl-number.txt。要使用的位字段符号定义在 <linux/ioctl.h>：

type(幻数)：8 位宽(\_IOC\_TYPEBITS)，参考ioctl-number.txt选择一个数，并在整个驱动中使用它。

number（序数）：顺序编号，8 位宽(\_IOC\_NRBITS)。

direction（数据传送的方向）：可能的值是 \_IOC\_NONE(没有数据传输)、\_IOC\_READ、\_IOC\_WRITE和 \_IOC\_READ|\_IOC\_WRITE（双向传输数据）。该字段是一个位掩码（两位），因此可使用 AND 操作来抽取 \_IOC\_READ 和 \_IOC\_WRITE。

size（数据的大小）：宽度与体系结构有关,ARM为14位.可在宏 \_IOC\_SIZEBITS 中找到特定体系的值。

<linux/ioctl.h> 中包含的 <asm/ioctl.h>定义了一些构造命令编号的宏：

```
_IO(type,nr)/*没有参数的命令*/
_IOR(type, nr, datatype)/*从驱动中读数据*/
_IOW(type,nr,datatype)/*写数据*/
_IOWR(type,nr,datatype)/*双向传送*/
/*type 和 number 成员作为参数被传递，并且 size 成员通过应用 sizeof 到 datatype 参数而得到*/
```

这个头文件还定义了用来解开这个字段的宏：

```
_IOC_DIR(nr)
_IOC_TYPE(nr)
_IOC_NR(nr)
_IOC_SIZE(nr)
```

具体的使用方法在实验中展示。

返回值

POSIX 标准规定：如果使用了不合适的 ioctl 命令号，应当返回-ENOTTY。这个错误码被 C 库解释为“不合适的设备 ioctl。然而，它返回-EINVAL仍是相当普遍的。

2013年（3）

2012年（61）

2011年（66）


2010年（27）

2009年（30）


2008年（23）

2007年（52）


我的朋友




小蜗牛快




cfm5538




jikaishi




shizhenc




pxy05215




李怀远



yan1990




wkm81018




xiousi


最近访客




apang199




appcount




zaichu




lhxzui




小蜗牛快




小尾巴鱼



erain\_30



hushup



wilfred\_

订阅

推荐博文

• linux 3.x的 通用时钟架构 ...

• SCN的相关解析

• Flash驱动学习

• 浅谈nagios之state type和 no...

• DB2（Linux 64位）安装教程...

• insert语句造成latch:library...

• 2014.06.13 网络公开课《让我...

• MySQL Slave异常关机的处理（...

• 巧用shell脚本分析数据库用户...

• 查询linux, HP-UX的cpu信息...

热词专题

• linux系统权限修复——学生误...

• Modbus协议使用

• linux

• busybox原理

• php环境搭建教程

预定义命令

有一些ioctl命令是由内核识别的，当这些命令用于自己的设备时，他们会在我们自己的文件操作被调用之前被解码。因此，如果你选择一个ioctl命令编号和系统预定义的相同时，你永远不会看到该命令的请求，而且因为ioctl 号之间的冲突，应用程序的行为将无法预测。预定义命令分为 3 类：

- （1）用于任何文件(常规，设备，FIFO和socket) 的命令
- （2）只用于常规文件的命令
- （3）特定于文件系统类型的命令

下列 ioctl 命令是预定义给任何文件，包括设备特定文件：

FIOCLEX：设置 close-on-exec 标志(File IOctl Close on EXec)。  
FIONCLEX：清除 close-no-exec 标志(File IOctl Not Close on EXec)。  
FIOQSIZE：这个命令返回一个文件或者目录的大小；当用作一个设备文件，但是，它返回一个 ENOTTY 错误。  
FIONBIO：“File IOctl Non-Blocking I/O”(在“阻塞和非阻塞操作”一节中描述)。

使用ioctl参数

在使用ioctl的可选arg参数时，如果传递的是一个整数，它可以直接使用。如果是一个指针，就必须小心。当用一个指针引用用户空间，我们必须确保用户地址是有效的，其校验(不传送数据)由函数 access\_ok 实现，定义在 <asm/uaccess.h>：

```
int access_ok(int type, const void *addr, unsigned long size);
```

第一个参数应当是 VERIFY\_READ（读）或VERIFY\_WRITE（读写）；addr 参数为用户空间地址，size 为字节数，可使用sizeof()。access\_ok 返回一个布尔值：1 是成功(存取没问题)和 0 是失败(存取有问题)。如果它返回假，驱动应当返回 -EFAULT 给调用者。

注意：首先，access\_ok不做校验内存存取的完整工作；它只检查内存引用是否在这个进程有合理权限的内存范围中，且确保这个地址不指向内核空间内存。其次，大部分驱动代码不需要真正调用 access\_ok，而直接使用put\_user(datum, ptr)和get\_user(local, ptr)，它们带有校验的功能，确保进程能够写入给定的内存地址，成功时返回 0，并且在错误时返回 -EFAULT。。

```
put_user(datum, ptr)
__put_user(datum, ptr)
get_user(local, ptr)
__get_user(local, ptr)
```

这些宏它们相对copy\_to\_user 和copy\_from\_user快，并且这些宏已被编写来允许传递任何类型的指针，只要它是一个用户空间地址。传送的数据大小依赖 prt 参数的类型，并且在编译时使用 sizeof 和 typeof 等编译器内建宏确定。他们只传送1、2、4或8 个字节。如果使用以上函数来传送一个大小不适合的值，结果常常是一个来自编译器的奇怪消息，如“conversion to non-scalar type requested”。在这些情况中，必须使用 copy\_to\_user 或者 copy\_from\_user。

\_\_put\_user和\_\_get\_user 进行更少的检查(不调用 access\_ok)，但是仍然能够失败如果被指向的内存对用户是不可写的，所以他们应只用在内存区已经用 access\_ok 检查过的时候。作为通用的规则：当实现一个 read 方法时，调用 \_\_put\_user 来节省几个周期，或者当你拷贝几个项时，因此，在第一次数据传送之前调用 access\_ok 一次。

权能与受限操作

Linux 内核提供了一个更加灵活的系统，称为权能（capability）。内核专为许可管理上使用权能并导出了两个系统调用 capget 和 capset，这样可以从用户空间管理权能，其定义在 <linux/capability.h> 中。对设备驱动编写者有意义的权能如下：

```
CAP_DAC_OVERRIDE /*越过在文件和目录上的访问限制(数据访问控制或 DAC)的能力。*/
CAP_NET_ADMIN /*进行网络管理任务的能力，包括那些能够影响网络接口的任务*/
CAP_SYS_MODULE /*加载或去除内核模块的能力*/
CAP_SYS_RAWIO /*进行“raw”（裸）I/O 操作的能力。例子包括存取设备端口或者直接和 USB 设备通讯*/
CAP_SYS_ADMIN /*截获的能力，提供对许多系统管理操作的途径*/
```

```
CAP_SYS_TTY_CONFIG /*执行 tty 配置任务的能力*/
```

在进行一个特权操作之前，一个设备驱动应当检查调用进程有合适的能力，检查是通过 `capable` 函数来进行的(定义在 `<linux/sched.h>`) 范例如下：

```
if (! capable (CAP_SYS_ADMIN))
    return -EPERM;
```

## 二、定位设备 (llseek实现)

`llseek`是修改文件中的当前读写位置的系统调用。内核中的缺省的实现进行移位通过修改 `filp->f_pos`，这是文件中的当前读写位置。对于 `lseek` 系统调用要正确工作，读和写方法必须通过更新它们收到的偏移量来配合。

如果设备是不允许移位的，你不能只制止声明 `llseek` 操作，因为缺省的方法允许移位。应当在你的 `open` 方法中，通过调用 `nonseekable_open` 通知内核你的设备不支持 `llseek`：

```
int nonseekable_open(struct inode *inode; struct file *filp);
```

完整起见，你也应该在你的 `file_operations` 结构中设置 `llseek` 方法到一个特殊的帮助函数 `no_llseek` (定义在 `<linux/fs.h>`)。具体的应用在试验程序中学习。

## 三、ioctl和llseek实验。

模块程序链接: `ioctl_and_llseek`

模块测试程序链接: `ioctl_and_llseek-test`

ARM9实验板的实验现象是：

```
[Tekkaman2440@SBC2440V4]#cd /lib/modules/
[Tekkaman2440@SBC2440V4]#insmod scull.ko scull_nr_devs=1
[Tekkaman2440@SBC2440V4]#cd /tmp/
[Tekkaman2440@SBC2440V4]#./scull_test2
open scull !
SCULL_IOCQQUANTUM-SCULL_IOCQQUANTUM : scull_quantum=10
SCULL_IOCQQUANTUM-SCULL_IOCQQUANTUM : scull_quantum=6
SCULL_IOCQQUANTUM : scull_quantum=6 --> 10
SCULL_IOCQQUANTUM : scull_quantum=10 --> 6
SCULL_IOCQSET-SCULL_IOCQSET : scull_qset=2
SCULL_IOCQSET-SCULL_IOCQSET : scull_qset=4
SCULL_IOCQSET : scull_qset=4 --> 2
SCULL_IOCQSET : scull_qset=2 --> 4
before reset : scull_quantum=6 scull_qset=4
close scull !
reopen scull !
reopen : scull_quantum=6 scull_qset=4
write code=6 i=20
write code=6 i=14
write code=6 i=8
write code=2
lseek scull SEEK_SET-->0 !
read code=6 i=20
read code=6 i=14
read code=6 i=8
read code=2
[0]=0 [1]=1 [2]=2 [3]=3 [4]=4
[5]=5 [6]=6 [7]=7 [8]=8 [9]=9
[10]=10 [11]=11 [12]=12 [13]=13 [14]=14
[15]=15 [16]=16 [17]=17 [18]=18 [19]=19
SCULL_IOCRESET
after reset : scull_quantum=4000 scull_qset=1000
close scull !
reopen scull !
write code=20
lseek scull SEEK_CUR-10-->10 !
read code=10
```

```
[0]=10 [1]=11 [2]=12 [3]=13 [4]=14
[5]=15 [6]=16 [7]=17 [8]=18 [9]=19
lseek scull SEEK_END-20-->0 !
read code=20
[0]=0 [1]=1 [2]=2 [3]=3 [4]=4
[5]=5 [6]=6 [7]=7 [8]=8 [9]=9
[10]=10 [11]=11 [12]=12 [13]=13 [14]=14
[15]=15 [16]=16 [17]=17 [18]=18 [19]=19
close scull !

[Tekkaman2440@SBC2440V4]#cat /proc/scullseq

Device 0: qset 1000, q 4000, sz 20
    item at c3dd3d74, qset at c3f54000
    0: c3e71000
[Tekkaman2440@SBC2440V4]#
```

阅读 (8801) | 评论 (2) | 转发 (37) |

上一篇：移植Linux2.6.22.2到博创2410-S(s3c2410A)

下一篇：[转载]中国最致命的薄弱环节！（一个机械类毕业生的心声）

0

相关热门文章

Linux设备驱动程序学习...	linux 常见服务端口	移植 ushare 到开发板
Linux设备驱动程序学习（1）-...	【ROOTFS搭建】busybox的httpd...	系统提供的库函数存在内存泄漏...
Linux设备驱动程序学习（2）-...	xmanager 2.0 for linux配置	linux虚拟机 求教
Linux设备驱动程序学习（3）-...	什么是shell	初学UNIX环境高级编程的，关于...
Linux设备驱动程序学习（4）-...	linux socket的bug??	chinaunix博客什么时候可以设...

给主人留下些什么吧！~~



kaikai10132012-07-13 19:11:02

仁兄，我用linux-3.2内核实验你的程序时，发现了以下错误，明明应用程序和驱动用的一样的CMD并且SCULL\_IOC\_MAGIC也一样，但是在驱动ioctl判断时却给出了“\_IOC\_TYPE(cmd) != SCULL\_IOC\_MAGIC”

回复 | 举报



tczf11282011-08-03 00:45:24

如果调用程序不传递arg参数，被驱动收到的 arg 值是未定义的？貌似是不传递cmd参数

回复 | 举报

评论热议

登录后评论。

[登录](#) [注册](#)

[关于我们](#) | [关于IT168](#) | [联系方式](#) | [广告合作](#) | [法律声明](#) | [免费注册](#)

Copyright 2001-2010 ChinaUnix.net All Rights Reserved 北京皓辰网域网络信息技术有限公司. 版权所有

感谢所有关心和支持过ChinaUnix的朋友们

京ICP证041476号 京ICP证060528号