


Linux编程之内存映射

4 回复 40 查看




(<https://www.shiyanlou.com/user/8490>) 实验楼管理员  (<https://www.shiyanlou.com/vip>)

5小时前


技术分享 (<https://www.shiyanlou.com/questions/?tag=技术分享>)

本文对Linux编程的内存映射做了较为详细的介绍~

 分享到微博

全部回答



实验楼管理员 (<https://www.shiyanlou.com/user/8490>)  (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

一.概述

内存映射是在调用进程的虚拟地址空间创建一个新的内存映射。

内存映射分为2种：

- **文件映射：**将一个普通文件的全部或者一部分映射到进程的虚拟内存中。映射后，进程就可以直接在对应的内存区域操作文件内容！
- **匿名映射：**匿名映射没有对应的文件或者对应的文件是虚拟文件(如：`/dev/zero`)，映射后会把内存分页全部初始化为0。

当多个进程映射了同一个内存区域时，它们会共享物理内存的相同分页。通过 `fork()` 创建的子进程也会继承父进程的映射副本！！

如果多个进程都会同一个内存区域操作时，会根据映射的特性，会有不同的行为。映射特征可分为私有映射和共享映射：

- **私有映射：**映射的内容对其他进程不可见。对于文件映射来说，某一个进程在映射内存中改变文件的内容不会反映到被映射的底层文件中。内核会使用copy-on-write(写时复制)技术来解决这个问题：只要有一个进程修改了分页中的内容，内核会为该进程重新创建一个新的分页，并将需要修改的内容复制到新分页中。
- **共享映射：**某一个进程对共享的内存区域操作都对其他进程可见！！对于文件映射，操作的内容会反映到底层文件中。

注意：进程执行`exec()`调用后，先前的内存映射会丢失，而`fork()`创建的子进程会继承父进程的，映射的特征(私有和共享)也会被继承。

异常信号：

- 当映射内存的属性设置只读时，如果进行写操作会产生SIGSEGV信号。
- 当映射内存的字节数大于被映射文件的大小，且大于该文件当前的内存分页大小时。如果访问的区域超过了该文件分页大小，会产生SIGBUS信号。

有点绕口，举个简单的例子：

假设内核维护的内存分页是4k(一般都是4k, 4096字节)，一个普通文件a.txt的大小是10字节。如果创建一个映射内存为4097字节，并映射该文件。此时，因为a.txt的大小用一个分页就可以完全映射，10字节远小于一个分页的4096字节，所以内核只会给它一个分页。内存地址是从0开始，0-9区间的内容对应a.txt文件的数据，我们也是可以访问10-4095的区间。但如果访问4096区间时，已经超过一个分页的大小了，此时会产生SIGBUS信号!!!

等会我们用个简单的例子演示下这2个异常。

二.函数接口

1.创建映射

```
#include <sys/mman.h>
void *mmap(void *addr, size_t length, int prot, int flags, int fd, off_t offset);
```

- `addr`：映射后要存放的虚拟内存地址。如果是NULL，内核会自动帮你选择。
- `length`：映射内存的字节数。
- `prot`：权限保护：`PROT_NONE`(无法访问)，`PROT_READ`(可读)，`PROT_WRITE`(可写)，`PROT_EXEC`(可执行)。
- `flags`：映射特征：`MAP_PRIVATE`(私有)，`MAP_SHARED`(共享)，`MAP_ANONYMOUS`。还有一些其他的可查询man手册。
- `fd`：要映射的文件描述符。
- `offset`：文件的偏移量，如果为0，且`length`为文件长度，代表映射整个文件。

2.解除映射

```
#include <sys/mman.h>
int munmap(void *addr, size_t length);
```

- `addr`：要解除内存的起始地址。如果`addr`不在刚刚映射区域的开始位置，解除一部分后内存区域可能会分成两半!!!
- `length`：要解除的字节数。

3.同步映射区

```
#include <sys/mman.h>
int msync(void *addr, size_t length, int flags);
```

- `addr`：要同步的内存起始地址。
- `length`：要同步的字节长度。
- `flags`：`MS_SYNC`(执行同步文件写入)，此操作内核会把内容直接写到磁盘。`MS_ASYNC`(执行异步文件写入)，此操作内核会先把内容写到内核的缓冲区，某个合适的时候再写到磁盘。

5小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>) <https://www.shiyanlou.com/vip>

(<https://www.shiyanlou.com/user/8490>)

三.文件映射实例

```

/** *
@file mmap_file.c
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <signal.h>
#include <unistd.h>
#include <sys/mman.h>

#define MMAP_FILE_NAME "a.txt"
#define MMAP_FILE_SIZE 10

void err_exit(const char *err_msg)
{
    printf("error:%s\n", err_msg);
    exit(1);
}

/* 信号处理器 */
void signal_handler(int signum)
{
    if (signum == SIGSEGV)
        printf("\nSIGSEGV handler!!!\n");
    else if (signum == SIGBUS)
        printf("\nSIGBUS handler!!!\n");
    exit(1);
}


int main(int argc, const char *argv[])
{
    if (argc < 2)
    {
        printf("usage:%s text\n", argv[0]);
        exit(1);
    }

    char *addr;
    int file_fd, text_len;
    long int sys_pagesize;

```

5小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>)  (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

接上面的代码。

```

/* 设置信号处理器 */
if (signal(SIGSEGV, signal_handler) == SIG_ERR)
    err_exit("signal()");
if (signal(SIGBUS, signal_handler) == SIG_ERR)
    err_exit("signal()");

if ((file_fd = open(MMAP_FILE_NAME, O_RDWR)) == -1)
    err_exit("open()");

/* 系统分页大小 */
sys_pagesize = sysconf(_SC_PAGESIZE);
printf("sys_pagesize:%ld\n", sys_pagesize);

/* 内存只读 */
//addr = (char *)mmap(NULL, MMAP_FILE_SIZE, PROT_READ, MAP_SHARED, file_fd, 0);

/* 映射大于文件长度, 且大于该文件分页大小 */
//addr = (char *)mmap(NULL, sys_pagesize + 1, PROT_READ | PROT_WRITE, MAP_SHARED, file_fd, 0);

/* 正常分配 */
addr = (char *)mmap(NULL, MMAP_FILE_SIZE, PROT_READ | PROT_WRITE, MAP_SHARED, file_fd, 0);
if (addr == MAP_FAILED)
    err_exit("mmap()");

/* 原始数据 */
printf("old text:%s\n", addr);

/* 越界访问 */
//addr += sys_pagesize + 1;
//printf("out of range:%s\n", addr);

/* 拷贝新数据 */
text_len = strlen(argv[1]);
memcpy(addr, argv[1], text_len);

/* 同步映射区数据 */
//if (msync(addr, text_len, MS_SYNC) == -1)
//    err_exit("msync()");

/* 打印新数据 */
printf("new text:%s\n", addr);


/* 解除映射区域 */
if (munmap(addr, MMAP_FILE_SIZE) == -1)
    err_exit("munmap()");

return 0;
}

```

5小时前



实验楼管理员 (<https://www.shiyanlou.com/user/8490>)  (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

1.首先创建一个10字节的文件:

```
$:dd if=/dev/zero of=a.txt bs=1 count=10
```

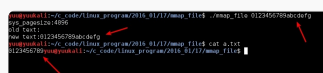
2.把程序编译运行后, 依次执行2写入:



可以看到本机的分页大小是4096字节。第一次写入9个字节, 原来用dd命令创建的文件为空, old text为空。第二次写入4个字节, 只覆盖了最前面的1234。

3.验证可访问现有分页的内存。

写入超过10字节的数据:



上面我们写入了17个字节，虽然64行的 `mmap()` 映射了 `MMAP_FILE_SIZE=10` 字节。但从输入new text可以看出，我们依然可以访问10字节后面的内存，因为该数据都在一个分页(4096)里面。cat查看a.txt后，只有前10个字节写入了a.txt。

4.验证SIGSEGV信号。

把64行注释调，58行打开，设置映射属性为只读，编译后访问：

```
root@kali:~/c_code/linux_program/2016_01/17/mmap_file_vxx# ./mmap_file_vxx
sys_page_size=4096
file_size=1028456789
SIGSEGV handler!!!
root@kali:~/c_code/linux_program/2016_01/17/mmap_file$
```

设置只读属性后，第77行有写操作。我们自定义的信号处理器就捕捉到了该信号。如果没有自定义信号处理器，终端就会输出 Segmentation fault

5.验证SIGBUS信号。

用61行的方法来映射内存。映射了一个分页大小再加1字节的内存，并放开72，73行的代码，让指针指向一个分页后的区域。编译后运行：

```
root@kali:~/c_code/linux_program/2016_01/17/mmap_file_vxx# ./mmap_file_vxx
sys_page_size=4096
file_size=1028456789
out of range
SIGBUS handler!!!
root@kali:~/c_code/linux_program/2016_01/17/mmap_file$
```

SIGBUS信号被自定义处理器捕捉到了。如果没有自定义信号处理器，终端就会输出Bus error

四.匿名映射

匿名映射有2种方式：

- 指定 `mmap()` 的flags参数为 `MAP_ANONYMOUS`，在linux上当指定这个值后会忽略fd参数的值。不过在有的UNIX上还需要把fd指定为-1。
- 把 `/dev/zero` 当做文件描述符打开，从 `/dev/zero` 读取数据时它会给你提供无穷无尽的0，向它写数据，它会丢弃。丢弃这点跟 `/dev/null` 一样，只是 `/dev/null` 不跟你提供数据。

匿名映射的使用跟上面的文件映射差不多。这里不再给例子。

文章地址：<http://www.linuxidc.com/Linux/2016-01/127558.htm>

作者：yuuyuu

5小时前

登录后才能回答问题哟~

我要提问

标签

Linux (<https://www.shiyanlou.com/questions/?tag=Linux>)

Python (<https://www.shiyanlou.com/questions/?tag=Python>)

C/C++ (<https://www.shiyanlou.com/questions/?tag=C/C++>)

实验环境 (<https://www.shiyanlou.com/questions/?tag=实验环境>)

技术分享 (<https://www.shiyanlou.com/questions/?tag=技术分享>)

功能建议 (<https://www.shiyanlou.com/questions/?tag=功能建议>)

课程需求 (<https://www.shiyanlou.com/questions/?tag=课程需求>)

Java (<https://www.shiyanlou.com/questions/?tag=Java>)

其他 (<https://www.shiyanlou.com/questions/?tag=其他>) SQL (<https://www.shiyanlou.com/questions/?tag=SQL>)

Hadoop (<https://www.shiyanlou.com/questions/?tag=Hadoop>) NodeJS (<https://www.shiyanlou.com/questions/?tag=NodeJS>)

常见问题 (<https://www.shiyanlou.com/questions/?tag=常见问题>) Web (<https://www.shiyanlou.com/questions/?tag=Web>)

Shell (<https://www.shiyanlou.com/questions/?tag=Shell>) PHP (<https://www.shiyanlou.com/questions/?tag=PHP>)

Git (<https://www.shiyanlou.com/questions/?tag=Git>) HTML (<https://www.shiyanlou.com/questions/?tag=HTML>)

HTML5 (<https://www.shiyanlou.com/questions/?tag=HTML5>) 信息安全 (<https://www.shiyanlou.com/questions/?tag=信息安全>)

网络 (<https://www.shiyanlou.com/questions/?tag=网络>) GO (<https://www.shiyanlou.com/questions/?tag=GO>)

NoSQL (<https://www.shiyanlou.com/questions/?tag=NoSQL>) 训练营 (<https://www.shiyanlou.com/questions/?tag=训练营>)

Android (<https://www.shiyanlou.com/questions/?tag=Android>) Ruby (<https://www.shiyanlou.com/questions/?tag=Ruby>)

Perl (<https://www.shiyanlou.com/questions/?tag=Perl>)

相关问题

Hive从概念到安装使用总结 (<https://www.shiyanlou.com/questions/2991>)

程序员必备基础知识：通信协议——Http、TCP、UDP (<https://www.shiyanlou.com/questions/2986>)

如何写出优美的 C 代码 (<https://www.shiyanlou.com/questions/2981>)

希望站内加入精华帖功能，或者我的关注，我的收藏等等 (<https://www.shiyanlou.com/questions/2967>)

Google Java编程风格指南 (<https://www.shiyanlou.com/questions/2966>)

动手做实验，轻松学IT。

实验楼-通过动手实践的方式学会IT技术。

公司简介 (<https://www.shiyanlou.com/aboutus>) 联系我们 (<https://www.shiyanlou.com/contact>) 常见问题 (<https://www.shiyanlou.com/faq#howtostart>)
我要开课 (<https://www.shiyanlou.com/labs>) 隐私协议 (<https://www.shiyanlou.com/privacy>) 会员条款 (<https://www.shiyanlou.com/terms>)
友情链接 (<https://www.shiyanlou.com/friends>)
站长统计 (http://www.cnzz.com/stat/website.php?web_id=5902315) 蜀ICP备13019762号 (<http://www.miibeian.gov.cn/>)



QQ群



微信



微博
(<http://weibo.com/shiyanlou2013>)