

Lab3

内核环境：(uname -r) 5.13.0-40-generic

第一部分：

主要结构体描述：

每个进程有一个 task_struct, 里面有一个 mm_struct, 里面有一个 vm_area_struct, 构成链表 (循环之类的), 而每个 vm_area_struct 都有一个指向 mm_struct 的指针。

```
struct mm_struct {
    struct vm_area_struct *mmap; // 虚拟区间 (VMA) 有序链表, 按照区间起始地址递增方式组织
    struct rb_root mm_rb; // VMA 红黑树根节点, 将进程所有的 VMA 记录到红黑树中, 以提高查找效率
    pgd_t *pgd; // 页全局目录
    int map_count; // VMA 数量
    struct rw_semaphore mmap_sem; // 读写信号量
    // ...
};

struct vm_area_struct {
    unsigned long vm_start; // VMA 起始地址
    unsigned long vm_end; // VMA 终止地址 (不包含本身)
    struct vm_area_struct *vm_next, *vm_prev; // 链表的后一个/前一个结点
    struct rb_node vm_rb; // 对应红黑树节点
    unsigned long vm_flags; // 权限, 在 mm.h 中定义
    // ...
}
```

伪代码流程：

pid 经过 find_get_pid->pid 结构体 经过 pid_task->得到 task_struct 结构体->得到 mm_struct 结构体,根据 mm 和传入的虚拟地址进行多级页表映射最后得到 page 结构体。
(另一种思路 (根据虚拟地址 find_vma ->得到虚拟块: vm_area_struct-> vm_mm)


```

[10391.216256] 找到物理页了！
[10391.216266] 可以执行到这里
[10391.216276] 此刻kernel_addr的值为:ffff927a73519000
[10391.216286] 此刻final_kernel_addr的值为:ffff927a73519dba
[10391.216296] 此刻读取到用户空间内C的值为:*
[10391.216367] Reading the proc file
[10391.216377] 用户想要得到的字节为的len为1024
[10391.216387] 文件系统的当前位置为0
[10391.216397] 写入user的字节为1
[10391.216530] 从用户空间传入的长度:23
[10391.216542] 把一字节内容2写入16637进程中7ffcbb13cdba地址的变量
[10391.216814] 找到物理页了！
[10391.216824] 可以执行到这里
[10391.216834] 此刻kernel_addr的值为:ffff927a73519000
[10391.216844] 此刻final_kernel_addr的值为:ffff927a73519dba
root@wf-virtual-machine:/home/wf/linux_files/linux_exp/lab3/SJTU#

```

可以看到,内核中输出为*,原因是42十进制对应的字符是*(ASCII码)

修改了mtest_test.c测试文件,不知道什么原因使用int c读入的char类型值不对,改成char就可以了。(21,22行)

第二部分:

根据实验要求文件的提示做即可。

调用alloc_page时用常规的GFP_KERNEL;查看remap_pfn_range的参数填写。

GFP_ATOMIC

用来从中断处理和进程上下文之外的其他代码中分配内存.从不睡眠.

GFP_KERNEL

内核内存的正常分配.可能睡眠.

GFP_USER

用来为用户空间页来分配内存;它可能睡眠.

结果:

```

root@wf-virtual-machine:/home/wf/linux_files/linux_exp/lab3/SJTU# cd "/home/wf/linux_files/linux_exp/lab3/SJTU/" && gcc maptest_test.c -o maptest_test && "/home/wf/linux_files/linux_exp/lab3/SJTU/"maptest_test
Listen to me say thanksroot@wf-virtual-machine:/home/wf/linux_files/linux_exp/lab3/SJTU#

```

感想:

这次实验需要自己查询很多API,而且发现有不同的方式(函数)来实现所需要的功能,很好的提高了自己根据需求查询函数和内核编程能力,同时也意识到自己能力尚且不足,需要和同学以及助教交流方且完成,独立工作的能力未来也很重要。

最后总结了做实验过程中的查找的比较好的资料供自己之后回

顾看:

<https://www.jianshu.com/p/89cd35010120>

<https://blog.csdn.net/prike/article/details/52722934>

dmesg 显示内核信息 dmesg -c (clear)

分页存储: https://blog.csdn.net/weixin_43914604/article/details/105907291

页, 页表, 页表项: <https://blog.csdn.net/HaoDaWang/article/details/78767830>

Mmap: <https://segmentfault.com/a/1190000014672234>

查看源码: <https://elixir.bootlin.com/linux/latest/source>

find_vma 函数功能描述: find_vma()函数根据一个属于某个进程的虚拟地址, 找到其所属的进程虚拟区间, 并返回相应的 vma_area_struct 结构体指针。

mm: 是进程整个用户空间的抽象, 也是总的控制结构, 一个进程只有一个 mm_struct 结构, 一个进程整个用户空间通常有若干离散的虚拟区间, 这些虚拟区间由 vm_area_struct 结构描述。

(为什么把进程的用户空间划分为一个个区间? 这是因为虚拟内存的来源不同, 有的来自可执行映像, 有的来自共享库, 而有的来自动态分配的内存区, 对不同的区间具有不同的访问权限, 有可能会有不同的操作, 所以需要把进程的用户空间分割管理, 并用虚存区处理函数)

在 linux 中, pfn 全称“page frame number”, 是物理内存区域编号

```
207
208 #define pte_page(pte) pfn_to_page(pte_pfn(pte))
209 #define mk_pte(page, pagemap)
```

C语言strncmp()函数: 比较字符串的前n个字符 (区分大小写)

strncmp() 用来比较两个字符串的前n个字符, 区分大小写, 其原型为: int strncmp (const char * str1, const char * str2, size_t n); 【参数】 str1, str2 为需要比较的 ...

```
3 #define pr_info(fmt, ...) \
4     printk(KERN_INFO pr_fmt(fmt), ##__VA_ARGS__)
```

输出的虚拟地址是 48 位的: addr: 7ffd8d3c9c97

<https://cn.etsoutdoors.com/858614-why-do-x86-64-systems-MQBRBCB>

unsigned long int

网页 学校 图片 视频 学术 更多

9,820,000 条结果 时间不限

unsigned long int 简介: unsigned long int, 在C语言中指**无符号长整型**的, 是整型 (整数 类型) 变量的一种。本类型与 unsigned long (“无符号长”) 是等价的, 即定义的时候 int (“整数”) 可以不写。

一个字是一个字节么

网页 学校 图片 视频 学术 更多

380,000,000 条结果 时间不限 自适应缩放

通常由**8个二进制位** (一个字节)来表示一个字符, 但也有少数计算机系统采用6个二进制的字符表示形式。

宏: <https://zhuanlan.zhihu.com/p/98470867>

```
void *p;

1  **在C语言中在任何时候都可以用void类型的指针来代替其他类型的指针, void指针可以指向任何数据类型的变量**
2  **如果要通过void指针去获取它所指向的变量值时候, 需要先将void指针强制类型转换成和变量名类型相匹配的数据类型指针后再进行操作;
3  指针的强类型转换:
4      void *p;
5      int *pa = (int *)p;
6      然后才能对原来的void指针指向的空间进行操作
7  任何类型的指针都可以赋值给void指针, 无需进行强制类型转换;
8      float f = 1.22f;
9      float *p1 = &f;
10     p = p1; //将float指针赋值给p
```



内核空间与物理地址之间有直接的映射关系：

<https://www.cnblogs.com/bizhu/archive/2012/10/09/2717303.html>

```
46  typedef __kernel_loff_t      loff_t;
47  #endif

88  typedef long long            __kernel_loff_t;

54  #define __SIZE_T__
55  typedef __kernel_size_t      size_t;

7  typedef unsigned long        __kernel_size_t;
```



分段 分页：

<https://blog.csdn.net/OOFFrankDura/article/details/84403741>

直接赋值 和 间接赋值：

<https://blog.csdn.net/shulianghan/article/details/121458154>

指针：

https://blog.csdn.net/qq_21583681/article/details/78572009#commentBox

问题：

char* buf = (char*) mmap(NULL, SIZE, PROT_READ, MAP_PRIVATE, fd, 0);需要 SIZE
ret = remap_pfn_range(vma, vma->vm_start, page_to_pfn(page), (unsigned long)index,
vma->vm_page_prot);需要一个 size((unsigned long)index)，二者有什么关系？
如何找到指针变量的地址？