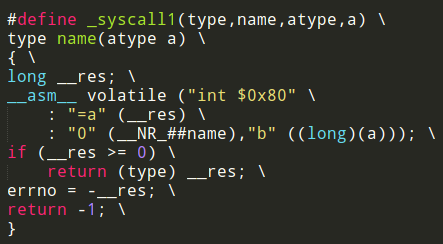
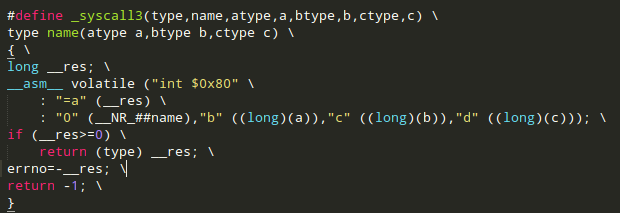
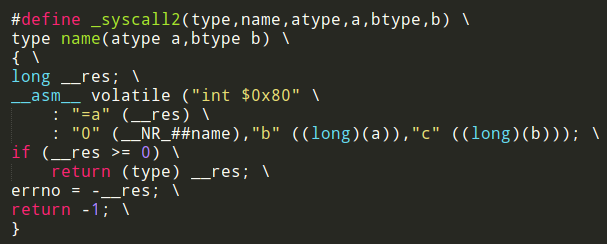
**实验1-2报告**

**1336101班 白广通 黄渝光**

1. 从Linux 0.11现在的机制看，它的系统调用最多能传递几个参数？你能想出办法来扩大这个限制吗？

答： 从linux-0.11/include/unistd.h中\_syscalln的宏展开我们可以知道，系统调用最多能够使用ebx，ecx，edx三个寄存器传递3个参数。

解决限制的方法：将需要传递的参数依次保存在具有特定结构的区间（如栈）中，并将这个区间的首地址作为一个参数，区间中保存的参数的个数作为另一个参数通过寄存器传递给系统调用函数。然后通过寄存器间接寻址方式便可以访问所有参数。#当然，这么做的话，必须要验证参数的合法性。

1. 用文字简要描述向Linux 0.11添加一个系统调用foo()的步骤。
   1. 首先，修改/usr/include/unistd.h，添加#define \_\_NR\_foo num，其中num为接下来使用的系统调用号。（要直接在linux-0.11中修改或修改后放到linux-0.11中）
   2. 其次，修改 include/linux/sys.h 在sys\_call\_table函数指针数组最后加入sys\_foo，并仿照上面给出其他系统调用格式加上对系统调用函数的声明:

extern rettype sys\_foo();

* 1. 然后修改 kernel/system\_call.s：

nr\_system\_calls = num

其中num为原值加1， 即系统调用总数目加1。

* 1. 接着在kernel中添加 foo.c 。若需要在内核态与用户态之间进行数据的交换，则需要包含include/asm/segment.h，其中有put\_fs\_XXX get\_fs\_XXX函数。同时，在foo.c实现系统调用sys\_foo()
  2. 最后修改kernel中的Makefile，将foo.c与内核中的其它代码编译链接到一起
  3. 实现系统调用的API:

#define \_\_LIBRARY\_\_

#include <unistd.h>

\_syscallN //宏展开系统调用，提供用户态下系统调用的接口（N代表参数数目，用以具体确定宏的名称）。