



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM

KHOA CÔNG NGHỆ THÔNG TIN

MÔN: **HỆ ĐIỀU HÀNH**

BÁO CÁO

Đồ án 2 Systemcall

GVHD: Trần Trung Dung

Lê Giang Thanh

Lê Quốc Hòa

Thực hiện: **Huỳnh Kim Ninh**

1612484

TP.HCM, 06/12/2018 7:57 CH

Mục lục

| | | |
|-----|---|----|
| 1. | Môi trường thực hiện | 3 |
| 2. | Tải mã nguồn linux-kernel (4.14.81) | 4 |
| 3. | Cài đặt syscall | 5 |
| 3.1 | Chuẩn bị mã nguồn | 5 |
| 3.2 | Biên dịch mã nguồn | 8 |
| 3.3 | Kiểm tra kết quả..... | 9 |
| 4. | Hook 2 syscall open và write..... | 11 |
| 4.1 | Chuẩn bị mã nguồn | 12 |
| 4.2 | Cài đặt module | 15 |
| 4.3 | Kết quả | 16 |
| 5. | Tài liệu tham khảo..... | 17 |

1. Môi trường thực hiện

- Hệ điều hành Ubuntu 16.04.5 Xenial – 64 bit
- Phiên bản kernel: [4.14.81](#)
- [Ubuntu16.04.5 Xenial Vmware \(VMDK\) 64-bit](#)
- VMware Player

2. Tải mã nguồn linux-kernel (4.14.81)

Truy cập <https://www.kernel.org/> và chọn phiên bản thích hợp tải về (khuyến khích chọn phiên bản mới)

Hoặc tải trực tiếp từ **Terminal** thông qua lệnh

```
wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.14.85.tar.xz
```

Giải nén và copy đến thư mục `/usr/src/`:

```
sudo tar -xvf linux-4.14.85.tar.xz -C /usr/src/
```

Ý nghĩa các thông số:

`tar` — Tar stores and extracts files from a tape or disk archive.

`-x` — extract files from an archive

`-v` — requested using the `-verbose` option, when extracting archives

`-f` — file archive; use archive file or device archive

`-C` — extract to the directory specified after it.(in this case `/usr/src/`)

3. Cài đặt syscall

Trước khi bắt đầu cài đặt syscall vào hệ thống, ta cần cài sẵn 1 số package hỗ trợ:

- libncurses5-dev
- bison
- flex
- libssl-dev

Ta sẽ tiến hành cài tất cả các package này thông qua lệnh sau:

```
sudo apt-get install gcc
sudo apt-get install libncurses5-dev
sudo apt-get install bison
sudo apt-get install flex
sudo apt-get install libssl-dev
sudo apt-get install libelf-dev
sudo apt-get update
sudo apt-get upgrade
```

3.1 Chuẩn bị mã nguồn

Tạo 1 thư mục có tên pnameid để chứa mã nguồn cho các file **pnameid.c**, **pnameid.h**, và **Makefile**

Nội dung các file mã nguồn như sau:

File pnameid.h:

```
asmlinkage long sys_pnametoid(char* name);
asmlinkage long sys_pidtoname(int pid, char* buf, int len);
```

File pnameid.c:

```
#include <linux/syscalls.h>
#include <linux/kernel.h>
#include <linux/sched.h>
#include <linux/init.h>
#include <linux/string.h>
#include "pnameid.h"
asmlinkage long sys_pnametoid(char* name){
    char buf[32];
    strncpy_from_user(buf, name, sizeof(buf));
    struct task_struct *task;
    int pid = -1;
    for_each_process(task){
        if(strcmp(task->comm,buf) == 0){
            pid = task->pid;
        }
    }
}
```

```

        return (long)pid;
    }
    asmlinkage long sys_pidtoname(int pid, char* buf, int len){
        struct task_struct *task;
        int n = -1;
        for_each_process(task){
            if(task->pid == pid){
                copy_to_user(buf, task->comm, len);
                n = strlen(task->comm);
            }
        }
        if (n == -1)
            return -1;

        if (len >= n)
            return 0;

        if (len < n)
            return n;
    }

```

File Makefile:

obj-y := pnameid.o

Thêm vào file Makefile của source kernel (**/usr/src/linux-4.14.81/Makefile**) thư mục `pnameid/`

core-y += kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/
pnameid/

Thêm hai system call mới vào bảng system call tại

/usr/src/linux-4.14.81/arch/x86/entry/syscalls/syscall_64.tbl

| | | | |
|-----|----|-----------|---------------|
| 333 | 64 | pnametoid | sys_pnametoid |
| 334 | 64 | pidtoname | sys_pidtoname |

```

syscall_64.tbl
/usr/src/linux-4.14.81/arch/x86/entry/syscalls

311 64 process_vm_writev sys_process_vm_writev
312 common kcmp sys_kcmp
313 common finit_module sys_finit_module
314 common sched_setattr sys_sched_setattr
315 common sched_getattr sys_sched_getattr
316 common renameat2 sys_renameat2
317 common seccomp sys_seccomp
318 common getrandom sys_getrandom
319 common memfd_create sys_memfd_create
320 common kexec_file_load sys_kexec_file_load
321 common bpf sys_bpf
322 64 execveat sys_execveat/ptregs
323 common userfaultfd sys_userfaultfd
324 common membarrier sys_membarrier
325 common mlock2 sys_mlock2
326 common copy_file_range sys_copy_file_range
327 64 preadv2 sys_preadv2
328 64 pwritev2 sys_pwritev2
329 common pkey_mprotect sys_pkey_mprotect
330 common pkey_alloc sys_pkey_alloc
331 common pkey_free sys_pkey_free
332 common statx sys_statx
333 64 pnametoid sys_pnametoid
334 64 pidtoname sys_pidtoname

#
# x32-specific system call numbers start at 512 to avoid cache impact
# for native 64-bit operation.
#
512 x32 rt_sigaction compat_sys_rt_sigaction
513 x32 rt_sigreturn sys32_x32_rt_sigreturn
514 x32 ioctl compat_sys_ioctl
515 x32 readv compat_sys_readv
516 x32 writev compat_sys_writev
517 x32 recvfrom compat_sys_recvfrom

Plain Text Tab Width: 8 Ln 342, Col 1 INS

.tbl
osboxes@osboxes:/usr/src/linux-4.14.81$ sudo gedit arch/x86/entry/syscalls/sysca
ll_64.tbl
[sudo] password for osboxes:

```

Thêm hai syscall `sys_pnametoid` và `sys_pidtoname` vào bảng `syscall_64.tbl`

Tiến hành thêm prototype của 2 syscall vào file `syscalls.h` tại

`/usr/src/linux-4.14.81/include/linux/syscalls.h`

```

asmlinkage long sys_pnametoid(char* name);
asmlinkage long sys_pidtoname(int pid, char* buf, int len);

```

```

syscalls.h [Read-Only] (/usr/src/linux-4.14.81/include/linux) - gedit
Open Save

asm linkage long sys_bpf(int cmd, union bpf_attr *attr, unsigned int size);

asm linkage long sys_execveat(int dfd, const char __user *filename,
                             const char __user *const __user *argv,
                             const char __user *const __user *envp, int flags);

asm linkage long sys_membarrier(int cmd, int flags);
asm linkage long sys_copy_file_range(int fd_in, loff_t __user *off_in,
                                     int fd_out, loff_t __user *off_out,
                                     size_t len, unsigned int flags);

asm linkage long sys_mlock2(unsigned long start, size_t len, int flags);

asm linkage long sys_pkey_mprotect(unsigned long start, size_t len,
                                   unsigned long prot, int pkey);
asm linkage long sys_pkey_alloc(unsigned long flags, unsigned long init_val);
asm linkage long sys_pkey_free(int pkey);
asm linkage long sys_statx(int dfd, const char __user *path, unsigned flags,
                          unsigned mask, struct statx __user *buffer);
asm linkage long sys_process_name(char* process_name);
asm linkage long sys_hello(void);
asm linkage long sys_pnametoid(char* name);
asm linkage long sys_pidtoname(int pid, char* buf, int len);
#endif

C/C++/ObjC Header Tab Width: 8 Ln 945, Col 1 INS
syscore_ops.h sysfs.h syslog.h sys_soc.h
osboxes@osboxes:/usr/src/linux-4.14.81$ gedit include/linux/syscalls.h

(gedit:12364): Gtk-WARNING **: Attempting to read the recently used resources fi
le at '/home/osboxes/.local/share/recently-used.xbel', but the parser failed: Fa
iled to open file '/home/osboxes/.local/share/recently-used.xbel': Permission de
nied.

```

Thêm khai báo hàm cài đặt vào syscalls.h

3.2 Biên dịch mã nguồn

Ngay tại thư mục linux-4.14.81/, ta thực hiện các lệnh sau để build mới kernel, quá trình này có thể kéo dài **2h** và tốn chừng **40GB ổ cứng**

```

sudo make menuconfig
sudo make
sudo make modules_install install

```

Sau khi cài đặt hoàn tất, ta cần restart máy lại để cập nhật bản kernel vừa build

```

shutdown -r now
uname -r

```


3.3 Kiểm tra kết quả

Ta sẽ cần file mã nguồn này để kiểm tra hai syscall pnametoid và pidtoname

File testpnameid.c

```
#include <stdio.h>
#include <linux/kernel.h>
#include <sys/syscall.h>
#include <unistd.h>
#include <string.h>

#define MAX_SIZE 32

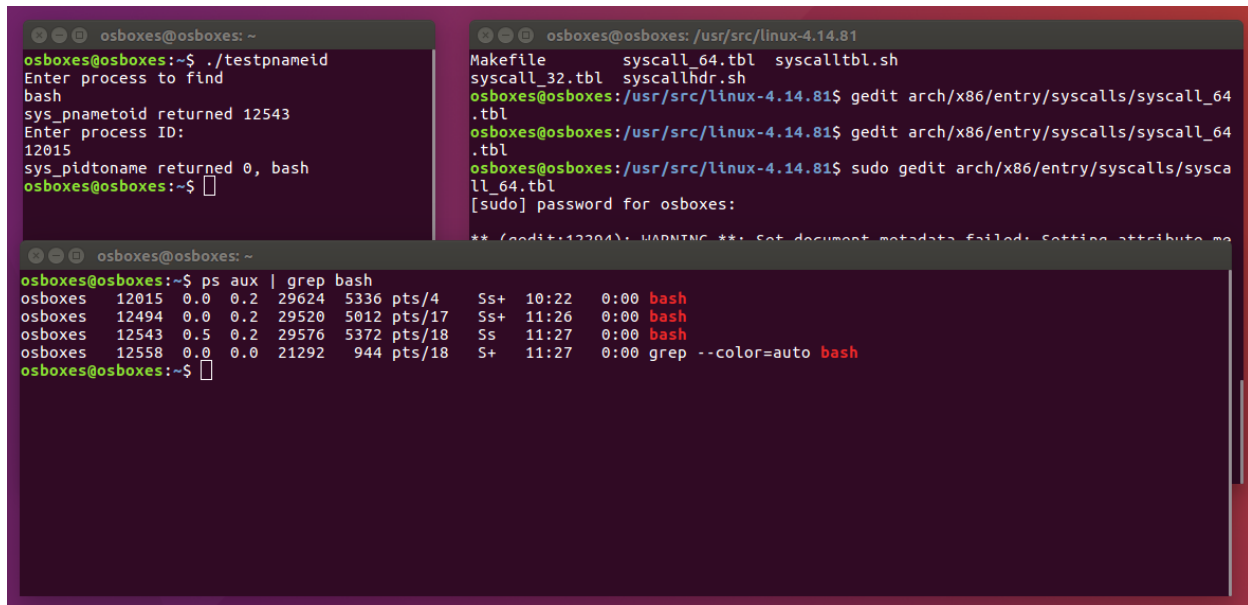
int main(){

    // Kiểm tra syscall pnametoid
    char name[32];
    puts("Enter process to find");
    scanf("%s",name);
    long int status = syscall(333, name);
    printf("sys_pnametoid returned %ld\n", status);

    // Kiểm tra syscall pidtoname
    int pid;
    char buff[MAX_SIZE];
    puts("Enter process ID: ");
    scanf("%d", &pid);
    long int status334 = syscall(334, pid, buff, MAX_SIZE);
    printf("sys_pidtoname returned %ld, %s\n", status334, buff);

    return 0;
}
```

Chạy chương trình lên và nhập vào `bash` ta sẽ được kết quả như sau:



```

osboxes@osboxes: ~
osboxes@osboxes:~$ ./testpnameid
Enter process to find
bash
sys_pnametoid returned 12543
Enter process ID:
12015
sys_pidtoname returned 0, bash
osboxes@osboxes:~$

osboxes@osboxes: /usr/src/linux-4.14.81
Makefile      syscall_64.tbl  syscalltbl.sh
syscall_32.tbl syscallhdr.sh
osboxes@osboxes: /usr/src/linux-4.14.81$ gedit arch/x86/entry/syscalls/syscall_64
.tbl
osboxes@osboxes: /usr/src/linux-4.14.81$ sudo gedit arch/x86/entry/syscalls/sysca
ll_64.tbl
[sudo] password for osboxes:
** (gedit:12304): WARNING **: Set document metadata failed: Setting attribute me
osboxes@osboxes: ~
osboxes@osboxes:~$ ps aux | grep bash
osboxes  12015  0.0  0.2  29624  5336 pts/4    Ss+  10:22   0:00  bash
osboxes  12494  0.0  0.2  29520  5012 pts/17    Ss+  11:26   0:00  bash
osboxes  12543  0.5  0.2  29576  5372 pts/18    Ss   11:27   0:00  bash
osboxes  12558  0.0  0.0  21292   944 pts/18    S+   11:27   0:00  grep --color=auto bash
osboxes@osboxes:~$
    
```

Syscall `pnametoid` nhận vào `bash` và trả ra PID cuối cùng được tìm thấy 12543

Ngược lại, syscall `pidtoname` nhận vào PID 12015 (cũng là 1 cái `bash` khác) và trả ra `bash`

4. Hook 2 syscall open và write

Đầu tiên ta cần phải lấy được địa chỉ của bảng syscall `sys_call_table` trong hệ điều hành. Có 2 nơi có thể tìm thấy được cái địa chỉ này: **/boot/System.map-4.14.81** và **/proc/kallsyms**. Tùy phiên bản kernel đang sử dụng mà ta sẽ chọn file phù hợp. (Với các bản **4.x trở lên** thì địa chỉ thật sự nằm trong **/proc/kallsyms**)

```

Terminal Terminal File Edit View Search Terminal Help
Open [icon] Save
#include <linux/unistd.h> /* sys_call_table __NR_* system call function indices
*/
#include <linux/fs.h> /* file open */

root@osboxes: /home/osboxes
362 ls
363 gedit hook.c
364 exit
365 gedit patch_syscall.c
366 gedit /proc/kallsyms
367 cat /boot/config-4.14.81
368 cat /boot/config-4.14.81 | grep CONFIG_VMSPLIT_*
369 cat /boot/config-4.14.81 | grep CONFIG_VMSPLIT
370 gedit /boot/config-4.14.81
371 ls
372 make
373 ls
374 insmod brute_forces_syscall.ko
375 rmmmod brute_forces_syscall
376 dmesg
377 cat /proc/kallsyms | grep sys_call_table
378 history
root@osboxes:/home/osboxes# cat /proc/kallsyms | grep sys_call_table
ffffffffb1a00180 R sys_call_table
ffffffffb1a01540 R ia32_sys_call_table
root@osboxes:/home/osboxes# cat /boot/System.map-4.14.81 | grep sys_call_table
ffffffff81e00180 R sys_call_table
ffffffff81e01540 R ia32_sys_call_table
root@osboxes:/home/osboxes#
  
```

Hai địa chỉ khác nhau ở 2 file khác nhau của bảng `sys_call_table`

Việc lấy địa chỉ của bảng **`sys_call_table`** có thể thông qua code cứng hoặc là brute force vùng nhớ ở kernel để lấy. Tuy nhiên để đơn giản thì chúng ta chỉ code cứng lưu lại **`0xffffffffb1a00180`** là được. (Lưu ý: mỗi lần restart lại máy thì địa chỉ trên sẽ **thay đổi** nên ta cũng sẽ cần **cập nhật thủ công** lại nếu có restart máy ảo)

Để có thể lấy được tên process đang chạy khi gọi đến syscall `open()`, `write()`, ta sẽ sử dụng đến macro `current`. Đây là 1 macro đặc biệt trỏ đến process đang thực thi bên trong kernel. Nó chứa đầy đủ các thông tin như ID, tên,...

Đối với syscall write(), ta sẽ cần viết thêm 1 hàm để lấy được tên file đang mở thông qua file descriptor (lấy từ tham số thứ nhất của write) và files_struct (lấy từ current->files). Hàm này là hàm getPathName() bên dưới.

4.1 Chuẩn bị mã nguồn

Tạo thư mục hooking để chứa mã nguồn cần thiết: **hook_open.c, hook_write.c, Makefile**

Lưu ý là chúng ta hoàn toàn có thể gộp chung thành 1 file hook.c duy nhất. Tuy nhiên để cho đơn giản, tránh việc sau khi hook 2 system call in thông báo báo lộn xộn thì ta sẽ hook lần lượt 2 system call này.

File hook_open.c, hook_write.c:

```
#include <linux/module.h> /* Needed by all kernel modules */
#include <linux/kernel.h> /* Needed for loglevels (KERN_WARNING,
KERN_EMERG, KERN_INFO, etc.) */
#include <linux/init.h> /* Needed for __init and __exit macros. */
#include <linux/unistd.h> /* sys_call_table __NR_* system call
function indices */
#include <linux/fs.h> /* filp_open */
#include <asm/paravirt.h> /* write_cr0 */
#include <linux/sched.h> /* need for current_id */
#include <linux/syscalls.h>
#include <linux/string.h>
#include <linux/fdtable.h> /*need for files_struct: current_files*/

unsigned long *syscall_table = (unsigned long *)0xffffffffb1a00180;

char *tmp = NULL;
char *pathname = NULL;

asmlinkage int (*original_write)(unsigned int, const char __user *,
size_t);

asmlinkage int (*original_open)(const char __user *, int);

int getPathName(unsigned int fd, struct files_struct *files){

    struct file *file;
    struct path *path;

    spin_lock(&files->file_lock);
    file = fcheck_files(files, fd);
    if (!file) {
        spin_unlock(&files->file_lock);
        return -ENOENT;
    }
}
```

```

    path = &file->f_path;
    path_get(path);
    spin_unlock(&files->file_lock);

    tmp = (char *)__get_free_page(GFP_KERNEL);

    if (!tmp) {
        path_put(path);
        return -ENOMEM;
    }

    pathname = d_path(path, tmp, PAGE_SIZE);
    path_put(path);

    if (IS_ERR(pathname)) {
        free_page((unsigned long)tmp);
        return PTR_ERR(pathname);
    }

    /* do something here with pathname */
    return 0;
}

asmlinkage int new_open(const char __user *pathname, int flags) {
    printk(KERN_INFO "[+] open() hooked.");

    char buf[32];

    strncpy_from_user(buf, pathname, sizeof(buf));
    printk(KERN_INFO "Process name: %s", current->comm);
    printk(KERN_INFO "File open: %s", buf);
    printk(KERN_INFO " ");

    return original_open(pathname, flags);
}

asmlinkage int new_write (unsigned int x, const char __user *y, size_t
size) {
    printk(KERN_INFO "[+] write() hooked.");
    char buf[32];
    int ret = getPathName(x, current->files);
    int writtenByte;

    strncpy_from_user(buf, y, sizeof(buf));

    printk(KERN_INFO "Process name: %s", current->comm);

```

```

    if (ret == 0)
    {
        printk(KERN_INFO "File name written: %s", pathname);
    }
    writtenByte = original_write(x, y, size);

    printk(KERN_INFO "Written byte: %d", writtenByte);
    printk(KERN_INFO " ");

    return writtenByte;
}

static int __init onload(void) {
    printk(KERN_WARNING "Hello world!\n");
    printk(KERN_INFO "Syscall table address: %p\n", syscall_table);

    if (syscall_table != NULL) {
        write_cr0 (read_cr0 () & (~ 0x10000));
        //original_write = (void *)syscall_table[__NR_write];
        //syscall_table[__NR_write] = &new_write;
        original_open = (void *)syscall_table[__NR_open];
        syscall_table[__NR_open] = &new_open;
        write_cr0 (read_cr0 () | 0x10000);
        printk(KERN_INFO "[+] onload: sys_call_table hooked\n");
        printk(KERN_INFO " ");
    } else {
        printk(KERN_INFO "[-] onload: syscall_table is NULL\n");
    }

    /*
     * A non 0 return means init_module failed; module can't be
    loaded.
     */
    return 0;
}

static void __exit onunload(void) {
    if (syscall_table != NULL) {
        write_cr0 (read_cr0 () & (~ 0x10000));
        //syscall_table[__NR_write] = original_write;
        syscall_table[__NR_open] = original_open;
        write_cr0 (read_cr0 () | 0x10000);
        free_page((unsigned long)tmp);
        printk(KERN_INFO "[+] onunload: sys_call_table unhooked\n");
    } else {
        printk(KERN_INFO "[-] onunload: syscall_table is NULL\n");
    }
}

```

```
    printk(KERN_INFO "Goodbye world!\n");
    printk(KERN_INFO " ");
}
```

```
module_init(onload);
module_exit(onunload);
```

Makefile:

```
obj-m += hook_open.o
obj-m += hook_write.o
```

```
all:
    make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
```

```
clean:
    make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

4.2 Cài đặt module

Ta tiến hành cài đặt module như sau:

```
root@osboxes: /home/osboxes/hooksing
osboxes@osboxes:~/hooking$ gedit hook.c
osboxes@osboxes:~/hooking$ gedit hook_write.c
osboxes@osboxes:~/hooking$ ls
hook_open.c hook_write.c Makefile
osboxes@osboxes:~/hooking$ sudo su
[sudo] password for osboxes:
root@osboxes: /home/osboxes/hooksing# make
make -C /lib/modules/4.14.81/build M=/home/osboxes/hooksing modules
make[1]: Entering directory '/usr/src/linux-4.14.81'
CC [M] /home/osboxes/hooksing/hook_open.o
/home/osboxes/hooksing/hook_open.c: In function 'new_open':
/home/osboxes/hooksing/hook_open.c:60:2: warning: ISO C90 forbids mixed declarations and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooksing/hook_open.c: In function 'new_write':
/home/osboxes/hooksing/hook_open.c:73:2: warning: ISO C90 forbids mixed declarations and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooksing/hook_open.c: In function 'onload':
/home/osboxes/hooksing/hook_open.c:102:28: warning: assignment makes integer from pointer without a cast [-Wint-conversion]
syscall_table[__NR_open] = &new_open;
^
/home/osboxes/hooksing/hook_open.c: In function 'onunload':
/home/osboxes/hooksing/hook_open.c:120:28: warning: assignment makes integer from pointer without a cast [-Wint-conversion]
syscall_table[__NR_open] = original_open;
^
CC [M] /home/osboxes/hooksing/hook_write.o
/home/osboxes/hooksing/hook_write.c: In function 'new_open':
/home/osboxes/hooksing/hook_write.c:60:2: warning: ISO C90 forbids mixed declarations and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooksing/hook_write.c: In function 'new_write':
/home/osboxes/hooksing/hook_write.c:73:2: warning: ISO C90 forbids mixed declarations and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooksing/hook_write.c: In function 'onload':
/home/osboxes/hooksing/hook_write.c:100:35: warning: assignment makes integer from pointer without a cast [-Wint-conversion]
syscall_table[__NR_write] = &new_write;
^
/home/osboxes/hooksing/hook_write.c: In function 'onunload':
/home/osboxes/hooksing/hook_write.c:119:35: warning: assignment makes integer from pointer without a cast [-Wint-conversion]
syscall_table[__NR_write] = original_write;
^
Building modules, stage 2.
MODPOST 2 modules
CC /home/osboxes/hooksing/hook_open.mod.o
LD [M] /home/osboxes/hooksing/hook_open.ko
CC /home/osboxes/hooksing/hook_write.mod.o
LD [M] /home/osboxes/hooksing/hook_write.ko
make[1]: Leaving directory '/usr/src/linux-4.14.81'
root@osboxes: /home/osboxes/hooksing#
```

4.3 Kết quả

Cài đặt module hook_open và xem kết quả

```

root@osboxes: /home/osboxes/hooks
/home/osboxes/hooks/hook_open.c:60:2: warning: ISO C90 forbids mixed declarati
ons and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooks/hook_open.c: In function 'new_write':
/home/osboxes/hooks/hook_open.c:73:2: warning: ISO C90 forbids mixed declarati
ons and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooks/hook_open.c: In function 'onload':
/home/osboxes/hooks/hook_open.c:100:35: warning: assignment makes integer from
pointer without a cast [-Wint-conversion]
syscall_table[__NR_open] = &new_open;
^
/home/osboxes/hooks/hook_open.c: In function 'onunload':
/home/osboxes/hooks/hook_open.c:120:28: warning: assignment makes integer from
pointer without a cast [-Wint-conversion]
syscall_table[__NR_open] = original_open;
^
CC [M] /home/osboxes/hooks/hook_write.o
/home/osboxes/hooks/hook_write.c: In function 'new_open':
/home/osboxes/hooks/hook_write.c:60:2: warning: ISO C90 forbids mixed declarati
ons and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooks/hook_write.c: In function 'new_write':
/home/osboxes/hooks/hook_write.c:73:2: warning: ISO C90 forbids mixed declarati
ons and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooks/hook_write.c: In function 'onload':
/home/osboxes/hooks/hook_write.c:119:35: warning: assignment makes integer fro
m pointer without a cast [-Wint-conversion]
syscall_table[__NR_write] = &new_write;
^
/home/osboxes/hooks/hook_write.c: In function 'onunload':
/home/osboxes/hooks/hook_write.c:119:35: warning: assignment makes integer fro
m pointer without a cast [-Wint-conversion]
syscall_table[__NR_write] = original_write;
^
Building modules, stage 2.
MODPOST 2 modules
CC /home/osboxes/hooks/hook_open.mod.o
LD [M] /home/osboxes/hooks/hook_open.ko
CC /home/osboxes/hooks/hook_write.mod.o
LD [M] /home/osboxes/hooks/hook_write.ko
make[1]: Leaving directory '/usr/src/linux-4.14.81'
root@osboxes: /home/osboxes/hooks# insmod hook_open.ko
root@osboxes: /home/osboxes/hooks# rmmod hook_open
root@osboxes: /home/osboxes/hooks#

2806.138686 [+] open() hooked.
2806.138690 Process name: bash
2806.138692 File open: .
2806.138693
2806.445082 [+] open() hooked.
2806.445088 Process name: thermald
2806.445089 File open: /sys/class/hwmon/hwmon0/temp1_in
2806.445091
2806.445949 [+] open() hooked.
2806.445952 Process name: thermald
2806.445953 File open: /sys/class/thermal/thermal_cooling_devic
2806.445955
2806.446058 [+] open() hooked.
2806.446058 Process name: thermald
2806.446060 File open: /sys/class/thermal/cooling_devic
2806.446061
2807.190241 [+] open() hooked.
2807.190242 Process name: rmmod
2807.190243 File open: /etc/ld.so.cache
2807.190243
2807.190257 [+] open() hooked.
2807.190257 Process name: rmmod
2807.190257 File open: /lib/x86_64-linux-gnu/libc.so.6
2807.190258
2807.190490 [+] open() hooked.
2807.190490 Process name: rmmod
2807.190491 File open: /lib/modules/4.14.81/modules.sof
2807.190491
2807.190534 [+] open() hooked.
2807.190535 Process name: rmmod
2807.190535 File open: /proc/cmdline
2807.190536
2807.190559 [+] open() hooked.
2807.190559 Process name: rmmod
2807.190559 File open: /lib/modules/4.14.81/modules.bui
2807.190560
2807.190600 [+] open() hooked.
2807.190601 Process name: rmmod
2807.190601 File open: /sys/module/hook_open/initstate
2807.190601
2807.190616 [+] open() hooked.
2807.190616 Process name: rmmod
2807.190616 File open: /sys/module/hook_open/holders
2807.190616
2807.190628 [+] open() hooked.
2807.190628 Process name: rmmod
2807.190628 File open: /sys/module/hook_open/refcnt
2807.190629
2807.190648 [+] onunload: sys_call_table unhooked
2807.190648 Goodbye world!
osboxes@osboxes:~$

```

Cài đặt module hook_write và xem kết quả

```

root@osboxes: /home/osboxes/hooks
char buf[32];
^
/home/osboxes/hooks/hook_open.c: In function 'new_write':
/home/osboxes/hooks/hook_open.c:73:2: warning: ISO C90 forbids mixed declarati
ons and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooks/hook_open.c: In function 'onload':
/home/osboxes/hooks/hook_open.c:100:35: warning: assignment makes integer from
pointer without a cast [-Wint-conversion]
syscall_table[__NR_open] = &new_open;
^
/home/osboxes/hooks/hook_open.c: In function 'onunload':
/home/osboxes/hooks/hook_open.c:120:28: warning: assignment makes integer from
pointer without a cast [-Wint-conversion]
syscall_table[__NR_open] = original_open;
^
CC [M] /home/osboxes/hooks/hook_write.o
/home/osboxes/hooks/hook_write.c: In function 'new_open':
/home/osboxes/hooks/hook_write.c:60:2: warning: ISO C90 forbids mixed declarati
ons and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooks/hook_write.c: In function 'new_write':
/home/osboxes/hooks/hook_write.c:73:2: warning: ISO C90 forbids mixed declarati
ons and code [-Wdeclaration-after-statement]
char buf[32];
^
/home/osboxes/hooks/hook_write.c: In function 'onload':
/home/osboxes/hooks/hook_write.c:100:35: warning: assignment makes integer fro
m pointer without a cast [-Wint-conversion]
syscall_table[__NR_write] = &new_write;
^
/home/osboxes/hooks/hook_write.c: In function 'onunload':
/home/osboxes/hooks/hook_write.c:119:35: warning: assignment makes integer fro
m pointer without a cast [-Wint-conversion]
syscall_table[__NR_write] = original_write;
^
Building modules, stage 2.
MODPOST 2 modules
CC /home/osboxes/hooks/hook_open.mod.o
LD [M] /home/osboxes/hooks/hook_open.ko
CC /home/osboxes/hooks/hook_write.mod.o
LD [M] /home/osboxes/hooks/hook_write.ko
make[1]: Leaving directory '/usr/src/linux-4.14.81'
root@osboxes: /home/osboxes/hooks# insmod hook_write.ko
root@osboxes: /home/osboxes/hooks# rmmod hook_write
root@osboxes: /home/osboxes/hooks#

3051.045435 Written byte: 8
3051.045435
3051.045450 [+] write() hooked.
3051.045451 Process name: gdbus
3051.045451 File name written: anon_inode:[eventfd]
3051.045452 Written byte: 8
3051.045452
3051.045454 [+] write() hooked.
3051.045454 Process name: gdbus
3051.045454 File name written: anon_inode:[eventfd]
3051.045455 Written byte: 8
3051.045455
3051.045455 [+] write() hooked.
3051.045455 Process name: gdbus
3051.045455 File name written: anon_inode:[eventfd]
3051.045456 Written byte: 8
3051.045456
3051.045508 [+] write() hooked.
3051.045509 Process name: gdbus
3051.045509 File name written: anon_inode:[eventfd]
3051.045510 Written byte: 8
3051.045510
3051.045560 [+] write() hooked.
3051.045562 Process name: gnome-terminal-
3051.045562 File name written: anon_inode:[eventfd]
3051.045564 Written byte: 8
3051.045564
3051.045643 [+] write() hooked.
3051.045645 Process name: gnome-terminal-
3051.045645 File name written: /dev/pts/1
3051.045670 Written byte: 1
3051.045670
3051.045725 [+] write() hooked.
3051.045728 Process name: bash
3051.045729 File name written: /dev/pts/17
3051.045734 Written byte: 1
3051.045735
3051.045759 [+] write() hooked.
3051.045761 Process name: gnome-terminal-
3051.045761 File name written: anon_inode:[eventfd]
3051.045762 Written byte: 8
3051.045762
3051.045767 [+] write() hooked.
3051.045768 Process name: gnome-terminal-
3051.045768 File name written: anon_inode:[eventfd]
3051.045769 Written byte: 8
3051.045769
3051.046472 [+] onunload: sys_call_table unhooked
3051.046472 Goodbye world!
osboxes@osboxes:~$

```


5. Tài liệu tham khảo

- [Adding a Hello World System Call to Linux Kernel](#) (Cài syscall trên ubuntu 16.04)
- [Syscall table hacking toy-example does not work on 64 bit](#) (Fix lỗi Paging request)
- [Basics of Making a Rootkit: From syscall to hook!](#)
- [How can I get a filename from a file descriptor inside a kernel module?](#) (Lấy đường dẫn file từ file descriptor)
- [how does current->pid work for linux?](#) (Cách sử dụng macro current)