

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ
ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ
ИНЖЕНЕРИИ

КУРСОВОЙ ПРОЕКТ
ЗАЩИЩЕН С ОЦЕНКОЙ
РУКОВОДИТЕЛЬ

ст.преп.

должность, уч. степень, звание

подпись, дата

Поляк М.Д.

инициалы, фамилия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К КУРСОВОМУ ПРОЕКТУ

СОЗДАНИЕ USB ДРАЙВЕРА

по дисциплине: ОПЕРАЦИОННЫЕ СИСТЕМЫ И СЕТИ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ 4331
ГР.

подпись, дата

инициалы, фамилия

Санкт-Петербург
2016

1 Цель работы

Цель работы: Знакомство с устройством ядра ОС Linux. Получение опыта разработки драйвера устройства.

2 Задание(4 вариант)

Добавление защиты от несанкционированного запуска операционной системы. Необходимо внести изменения в процесс загрузки ядра Linux, добавив проверку наличия подключенного через интерфейс USB flash-накопителя с заданным серийным номером. Если в процессе загрузки операционной системы нужный flash-накопитель подключен к одному из портов USB, то операционная система успешно загружается в штатном режиме. Если flash-накопитель с нужным серийным номером отсутствует, необходимо приостановить загрузку операционной системы и предоставить пользователю три попытки подключить нужный flash-накопитель. Если пользователь три раза подключает flash-накопитель с неправильным серийным номером, произвести выключение компьютера. Если же пользователь подключит нужный flash-накопитель, продолжить загрузку операционной системы в штатном режиме.

3 Техническая документация

1. Сборка проекта:

Скачиваем файлы с репозитория на github при помощи команды:

```
git clone https://github.com/andreytantsura/usbdriver.git
```

2. Добавление в автозагрузку:

Шаг 1: Собираем драйвер (test.ko) с помощью запуска команды make.

Шаг 2: Копируем файл в папку /lib/modules/версия ядра/test.ko.

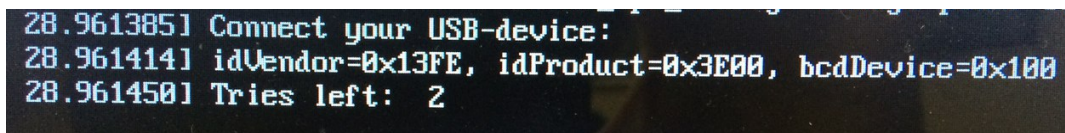
Шаг 3: Добавляем драйвер в автозагрузку с помощью команды depmod test.ko.

Шаг 4: Отключаем флеш-устройство при загрузке системы.

Шаг 5: Перезгружаем систему.

Шаг 6: Все работает, система требует флешку для запуска.

4 Скриншоты



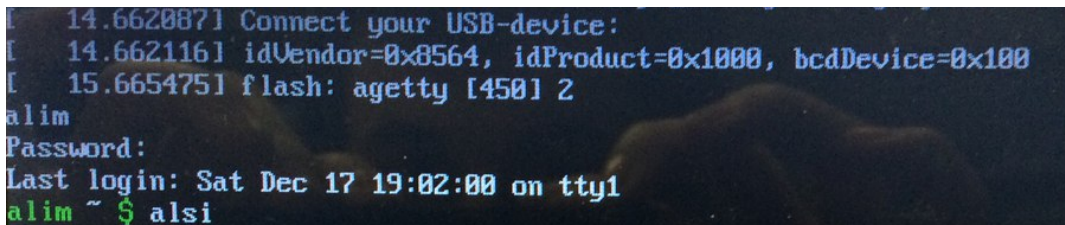
```
28.961385] Connect your USB-device:
28.961414] idVendor=0x13FE, idProduct=0x3E00, bcdDevice=0x100
28.961450] Tries left: 2
```

Рис. 1: Вставка неправильной флешки



```
28.961385] Connect your USB-device:
28.961414] idVendor=0x13FE, idProduct=0x3E00, bcdDevice=0x100
28.961450] Tries left: 2
41.629573] flash: disconnect
46.128066] Connect your USB-device:
46.128095] idVendor=0x13FE, idProduct=0x3E00, bcdDevice=0x100
46.128129] Tries left: 1
55.348143] flash: disconnect
61.174750] Connect your USB-device:
61.174779] idVendor=0x13FE, idProduct=0x3E00, bcdDevice=0x100
61.174813] Tries left: 0
61.174836] Your computer will reboot in 3...
62.177980] Your computer will reboot in 2...
63.181348] Your computer will reboot in 1...
```

Рис. 2: Вставка неправильной флешки третий раз перезагрузка компьютера



```
14.662087] Connect your USB-device:
14.662116] idVendor=0x8564, idProduct=0x1000, bcdDevice=0x100
15.665475] flash: agetty [450] 2
alim
Password:
Last login: Sat Dec 17 19:02:00 on tty1
alim ~ $ alsi
```

Рис. 3: Вставка корректной флешки

5 Выводы

В процессе выполнения данной курсовой работы мною были получены знания и навыки, необходимые для работы с ядром ОС Linux, а так же знания и навыки в разработке драйверов устройств.

6 Приложение

test.c:

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/usb.h>
#include <linux/sched.h>
#include <linux/kthread.h>
#include <linux/types.h>
#include <linux/tty.h>
#include <linux/version.h>
#include <linux/delay.h>
#include <linux/reboot.h>

struct task_struct *tAgetty;
struct task_struct *task;
bool stopThread = true;
bool isTry = true;
static int param = 1;
module_param( param, int, 0 );
int countTry = 3;

static int thread_agetty_uninterruptible( void * data)
{
    // основной цикл потока
    while(stopThread)
    {
        for_each_process(task)
        {

            if (strcmp(task->comm, "agetty") == 0 && task->state == TASK_INTERRUPTIBLE)
            {
                ssleep(1);
                printk(KERN_ERR "tty: %s [%d] %u \n", task->comm , task->pid, (u32)task->state);
                task->state = TASK_UNINTERRUPTIBLE;
            }
        }
    }
    if (countTry <= 0)
    {
        printk(KERN_ERR "Your computer will reboot in 3...\n");
        ssleep(1);
        printk(KERN_ERR "Your computer will reboot in 2...\n");
        ssleep(1);
    }
}
```

```

printk(KERN_ERR "Your computer will reboot in 1...\n");
ssleep(1);
kernel_restart(NULL);
}

return -1;
}

static int pen_probe(struct usb_interface *interface, const struct usb_dev
{
struct usb_device *dev = interface_to_usbdev(interface);
printk(KERN_ERR "Connect your USB-device:\n");

printk( KERN_ERR "idVendor=0x%hX, idProduct=0x%hX, bcdDevice=0x%hX\n",
dev->descriptor.idVendor,
dev->descriptor.idProduct, dev->descriptor.bcdDevice );

if (isTry)
{
if (dev->descriptor.idVendor == 0x8564 && dev->descriptor.idProduct == 0x1
{
stopThread = false;
isTry = false;
ssleep(1);
for_each_process(task)
{
if (strcmp(task->comm, "agetty") == 0 && task->state == TASK_UNINTERRUPTIB
{
printk(KERN_ERR "flash: %s [%d] %u \n", task->comm , task->pid, (u32)task-
task->state = TASK_INTERRUPTIBLE;
}
}
} else {

countTry--;
printk(KERN_ERR "Tries left:  %i \n", countTry);
if (countTry <= 0)
{
stopThread = false;
}
}
}
}

```

```

return 0;
}

static void pen_disconnect(struct usb_interface *interface)
{
    printk(KERN_ERR "flash: disconnect\n");
}

static struct usb_device_id pen_table[] =
{
    { USB_DEVICE(0x13FE, 0x3E00) },
    { USB_DEVICE(0x8564, 0x1000) },
    {} /* Terminating entry */
};

MODULE_DEVICE_TABLE (usb, pen_table);

static struct usb_driver pen_driver =
{
    .name = "driver",
    .probe = pen_probe,
    .disconnect = pen_disconnect,
    .id_table = pen_table,
};

static int __init pen_init(void)
{
    printk("init: start\n");

    // поток блокирования tty
    tAgetty = kthread_create( thread_agetty_uninterruptible, NULL, "agetty_uni

    if (!IS_ERR(tAgetty))
    {
        printk(KERN_INFO "thread: %s start\n", tAgetty->comm);
        wake_up_process(tAgetty);
    }
    else
    {
        printk(KERN_ERR "thread: agetty_uninterruptible error\n");
        WARN_ON(1);
    }
}

```

```
return usb_register(&pen_driver);  
}  
  
static void __exit pen_exit(void)  
{  
usb_deregister(&pen_driver);  
}  
  
module_init(pen_init);  
module_exit(pen_exit);  
  
MODULE_LICENSE("GPL");  
MODULE_AUTHOR("none");  
MODULE_DESCRIPTION("USB Pen Info Driver");
```