

ГУАП

Кафедра № 43

ОТЧЁТ  
ЗАЩИЩЁН С ОЦЕНКОЙ  
ПРЕПОДАВАТЕЛЬ

ст.преп.

М.Д. Поляк

должность, уч. степень, звание

подпись, дата

инициалы, фамилия

ОТЧЁТ О КУРСОВОЙ РАБОТЕ

Добавление защиты от несанкционированного запуска  
операционной системы.

по ОПЕРАЦИОННЫМ СИСТЕМАМ

РАБОТУ ВЫПОЛНИЛ:

СТУДЕНТ ГР. 4336

Т.М.Пилипчук

подпись, дата

инициалы, фамилия

Санкт-Петербург, 2017

# 1 Цель работы

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

## 2 Задание

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, отобразить на экране предупреждение и таймер с обратным отсчетом (30 секунд), загрузка операционной системы при этом приостанавливается. По истечении обратного отсчета таймера должно происходить автоматическое выключение компьютера. При подключении к любому из USB-портов нужного flash-накопителя во время обратного отсчета таймера, таймер должен останавливаться, после чего операционная система должна загрузиться в штатном режиме.

## 3 Техническая документация

Сборка и добавление в автозагрузку:

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

Шаг 2: Копируем драйвер (test.ko) с помощью команды "cp test.ko /usr/lib/modules/(версия ядра)/"

Шаг 3: Добавим в автозагрузку следующей командой "echo 'test' > /etc/modulesload.d/

Шаг 4: Отключаем флеш-устройство при загрузке системы.

Шаг 5: Перезагружаем систему.

Шаг 6: При загрузке система требует вставить флешку с определённым серийным номером, её необходимо вставить в течении 30 секунд иначе система выключится.

## 4 Скриншоты

Скриншоты выполнения работы программы:

На Рис. 1 изображена Реакция, когда был вставлен флэш-накопитель с нужным серийным номером.

На Рис. 2 изображена реакция, когда был вставлен флэш-накопитель с ложным серийным номером.

```
[ 45.630370] USB Connected: idVendor=0x781, idProduct=0x5567, Serial=4C5320000
40915103484
[ 46.640859] Key USB device connected

Arch Linux 4.8.13-1-ARCH (tty1)

albert login: [ 108.710428] USB device disconnected

Arch Linux 4.8.13-1-ARCH (tty1)

albert login:
```

Рис. 1:

```
Arch Linux 4.8.13-1-ARCH (tty1)

albert login: [ 24.880584] tty: agetty [284] 1
[ 24.880584] Waiting key USB device. 3 attempts
[ 33.076956] USB Connected: idVendor=0x8564, idProduct=0x1000, Serial=21B7FG8U
6SSLOFFE
[ 33.083660] Tries left: 2
[ 37.067314] USB device disconnected
```

Рис. 2:

## 5 Заключение

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

```
#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 \nWaiting key USB device. %d attempts\n",
                    task->state = TASK_UNINTERRUPTIBLE;
                }
            }
        }
        if (countTry <= 0)
        {
            printk(KERN_ERR "Reboot in 3...\n");
            ssleep(1);
            printk(KERN_ERR "Reboot in 2...\n");
            ssleep(1);
            printk(KERN_ERR "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 "USB Connected: idVendor=0x%hX, idProduct=0x%hX, Serial=%
        dev->descriptor.idVendor,
        dev->descriptor.idProduct, dev->serial );

        if (isTry)

```

```

{
if (dev->descriptor.idVendor == 0x0951 && dev->descriptor.idProduct == 0x1
{
stopThread = false;
isTry = false;
ssleep(1);
printk( KERN_ERR "Key USB device connected\n");

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)tas
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 "USB device disconnected\n");
}

static struct usb_device_id pen_table[] =
{
{ USB_DEVICE(0x0951, 0x1603) },
{}
};

static struct usb_driver pen_driver =
{

```

```

.name = "usb_auth",
.probe = pen_probe,
.disconnect = pen_disconnect,
.id_table = pen_table,
};

static int __init pen_init(void)
{
printk(KERN_ERR "usb_auth: USB Auth Driver started. 3 attempts\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 Auth Driver");

```