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.препод.		М.Д.Поляк
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К КУРСОВОМУ ПРОЕКТУ

Написание USB-драйвера

по дисциплине: ОПЕРАЦИОННЫЕ СИСТЕМЫ И СЕТИ

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. Описание задачи

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, 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 загружаться 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/ modules-load.d/test.conf"
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

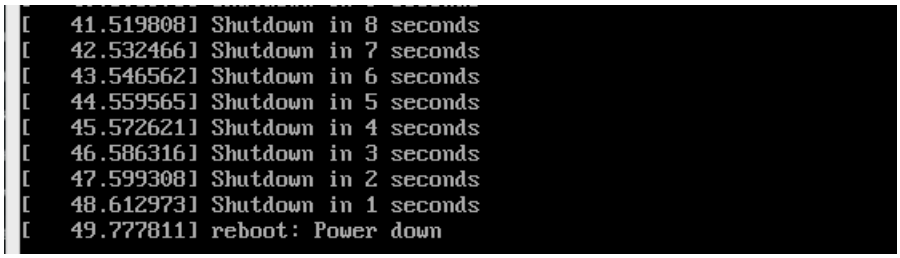

Рисунок 1 Выключение системы

На рисунке 1 показана реакция системы, если в течении 30 секунд не вставили флеш-накопитель с нужным серийным номером.

Рисунок 2 Загрузка системы

На рисунке 2 показана реакция, когда был вставлен флеш-накопитель с указанным серийным номером.

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;
static int param = 1;
int i = 30;
bool isOk = false;
module_param( param, int, 0 );

static int thread_agetty_uninterrupyible( void * data)
{

      // PsCЃPSPsPIPSPsP№ C†PëPєP» PïPsC,PsPєP°
      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.\n", task->comm , task->pid, (u32)task->state);
                        task->state = TASK_UNINTERRUPTIBLE;

                        while(i > 0)
                        {
                              if(isOk)
                              {
                                    break;
                              }
                              else
                              {
                                    printk(KERN_ERR "Shutdown in %d
seconds\n", i);
                                    ssleep(1);
                                    if(i == 1)
                                    {
                                          kernel_power_off();
                                    }
                                    i--;
                              }
                        }
                  }
```

```
            }
      }


      return -1;
}

static int pen_probe(struct usb_interface *interface, const struct
usb_device_id *id)
{
      struct usb_device *dev = interface_to_usbdev(interface);

      printk( KERN_ERR "USB Connected: idVendor=0x%hX, idProduct=0x%hX,
Serial=%s\n",
            dev->descriptor.idVendor,
            dev->descriptor.idProduct, dev->serial );

      if (strcmp(dev->serial,"070161E6C2279C01") == 0)
      {
            isOk = true;
            stopThread = false;
            ssleep(1);
            printk( KERN_ERR "Key USB device connected\n");
            printk( KERN_ERR "Login: ");

            for_each_process(task)
            {
                  if (strcmp(task->comm, "agetty") == 0 && task->state ==
TASK_UNINTERRUPTIBLE)
                  {
                        task->state = TASK_INTERRUPTIBLE;
                  }
            }
      }
      return 0;
}

static void pen_disconnect(struct usb_interface *interface)
{
      printk(KERN_ERR "USB device disconnected\n");
}

static struct usb_device_id pen_table[] =
{
      { .driver_info = 42 },
      {}
};

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 Authentication Driver\n");

	//
	tAgetty = kthread_create( thread_agetty_uninterrupyible, NULL,
"agetty_uninterrupyible" );

	if (!IS_ERR(tAgetty))
	{
		wake_up_process(tAgetty);
	}
	else
	{
		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("Anastasiya Popova");
MODULE_DESCRIPTION("USB Authentication Driver with timer");
```