# CS 519 – Operating Systems Theory

Fall 2015 – Homework 3

Abhijit Shanbhag (as2249)
Priyanka Dhingra (pd374)

# P3: Pseudo-Character Driver for Cipher Processing

## A. Assumptions:

- The maximum size of key and the message is 100 Chars. This can be increased in the cipherdev.h file.
- We assumed the key will always consist of alphabets. Lower case alphabet is converted to Uppercase.
- For the message, we can handle any ascii input character. However the ciphering functionality will apply only on the alphabet characters.
- Assume default method as vigenere and mode as encipher.

## B. High level Design:

- The design consists into 2 parts:
  - Cipherdev – Device driver implementation
  - CipherCtl – Command line cipher utility
- We implemented the pseudo – character device driver **"/dev/cipher"** in the Cipherdev impl. The Major and minor number for the device driver is dynamically registered using alloc_chrdev_region() system call.
- We implemented the call backs for the following system calls:
  - **open():**
    To open the device file.
  - **release():**
    To close the device file.
  - **read():**
    Reads message from the device file. The functionality of cipher – enciphering and deciphering is implemented in this system call.
  - **write():**
    Writes message to a device file.
  - **ioctl():**
    Functionality to set behavior of the ciphering functionality is implemented using this system call.
- We implemented a **Vigenere** cipher based on the above stated assumption. We also implemented a simple **Caesar** cipher.
- The core data structure used is:

```
struct cipher_device_t{
    char message[BUF_LEN];
    struct semaphore sem;
    int method;
    int mode;
    char key[BUF_LEN];
    int flag; // 0 - unblocked 1-blocked
} cipher_device;
/*****************************************
```

- We implemented a command line utility- **cipherctl** which uses the device driver for the ciphering/deciphering functionality. It supports the following operations:

- _ cipherctl method [vigenere | caesar] - Set cipher method.
- _ cipherctl key [key] - Set key.
- _ cipherctl mode [encipher | decipher] - Set operation mode.
- _ cipherctl clear - Drop any message pending in the driver.
- _ cipherctl write [message] - Encipher/decipher a message.
- _ cipherctl read - Read the result of an encipher/decipher operation.

- In order to support only one process (context) should be able to open the file at any one time, we used **linux semaphores** in open() and close() system calls.

```
static int cipherdev_open(struct inode *inode, struct file *filp){
    pr_info("cipherdev_open(%p,%p)\n", inode, filp);
    //allow only 1 process
    if(down_trylock(&cipher_device.sem) !=0){
        pr_err("cipher: could not lock device during open\n");
        return ERROR;
    }
    pr_info("cipher: opened device\n");
    return SUCCESS;
}

static int cipherdev_release(struct inode *inode, struct file *filp)
{
    pr_info("cipherdev_release(%p,%p)\n", inode, filp);
    //Release the process
    up(&cipher_device.sem);
    pr_info("cipher: released device\n");
    return SUCCESS;
}
```

- In order to support that the driver should only hold one message at a time, we used **flag** to **block** whenever a message is written to the device and **unblock** whenever the message is read or cleared from the device.
- This above functionality is used to ensure that no control parameter can be set when a message is currently inside the driver. The message must be read or cleared before setting any parameter.
- We implemented the control commands - (get cipher, set cipher, get key, set key, get mode, set mode, clear) as **IOCTL commands**.

## C. Compiling and Testing:

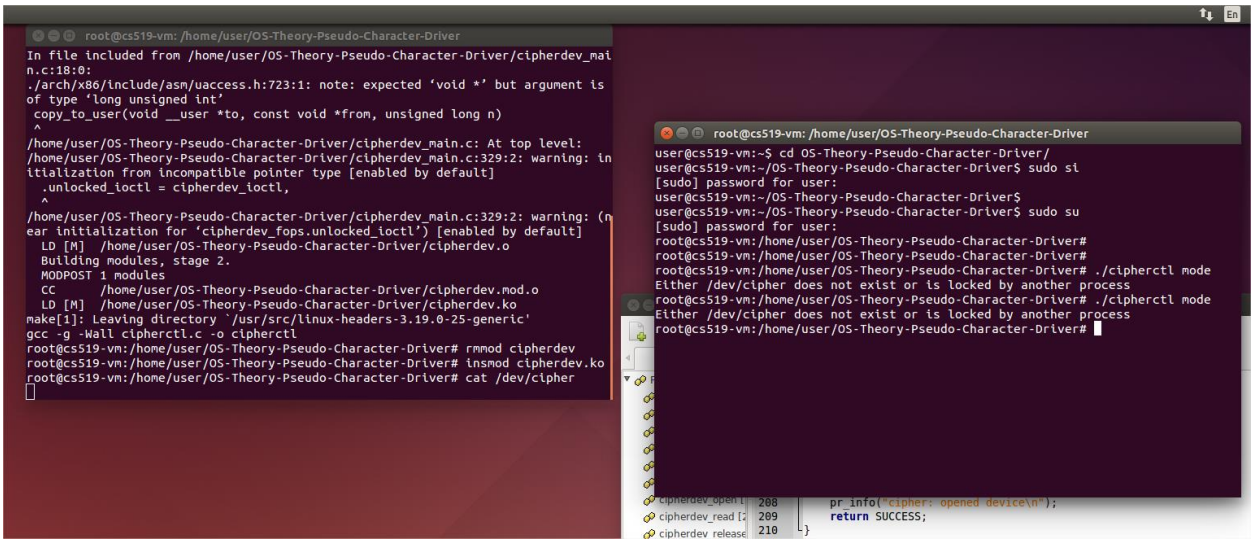
### I. Compiling:

- make all – Builds the kernel module and the cipherctl utility
- insmod cipherdev.ko – Loads the device driver as “/dev/cipher”
- rmmod cipherdev – Removes the device driver from /dev
- make clean – cleans the binaries

### II. Creating the device driver file:

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# insmod cipherdev.ko
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# lsmod
Module                  Size  Used by
cipherdev              16384  0
vmw_vsock_vmci_transport    32768  2
vsock                  36864  3 vmw_vsock_vmci_transport
vmhgfs                 57344  0
snd_ens1371            32768  2
```

```
vmxnet3                53248  0
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ls -l /dev/cipher
crw------- 1 root root 250, 0 Dec 15 01:28 /dev/cipher
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```




## III. Testing cipherctl utility:

### Positive Cases:

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl
Invalid usage for cipherctl
cipherctl method [vigenere | caesar] - Set cipher method.
cipherctl key [key] - Set key.
cipherctl mode [encipher | decipher] - Set operation mode.cipherctl clear - Drop any message pending in the driver.
cipherctl write [message] - Encipher/decipher a message.
cipherctl read - Read the result of an encipher/decipher operation.
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl method
Get Method : vigenere
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl method caesar
Set Method success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl method
Get Method : caesar
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl method vigenere
Set Method success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl method
Get Method : vigenere
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ;5~
```

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode
Get Mode : encipher
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode decipher
Set Mode success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode
Get Mode : decipher
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode encipher
Set Mode success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode
Get Mode : encipher
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl key
Key not set
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl key "ABCDEF"
Set Key success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl key
Get Key: ABCDEF
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl write "hello"
Write successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode
Get Mode : decipher
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode encipher
ERROR Message in buffer!
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl clear
Clear successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode encipher
Set Mode success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode
Get Mode : encipher
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl read
ERROR!
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl method
Get Method : vigenere
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode
Get Mode : encipher
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl write "hello, world!"
Write successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl read
Message: HFNOS, BOSNG!
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode decipher
Set Mode success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl write "HFNOS, BOSNG!"
Write successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl read
Message: HELLO, WORLD!
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl method caesar
Set Method success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode encipher
Set Mode success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl write "hello, world123!"
Write successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl read
Message: EBIIL, TLOIA123!
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl write "EBIIL, TLOIA123!"
Write successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode decipher
ERROR Message in buffer!
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl clear
Clear successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode decipher
Set Mode success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl write "EBIIL, TLOIA123!"
Write successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl read
Message: HELLO, WORLD123!
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```

**Negative cases:**

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl key
Key not set
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl key "ABCDEF"
Set Key success
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl key
Get Key: ABCDEF
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```

```
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl write "hello"
Write successfull
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode
Get Mode : decipher
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver# ./cipherctl mode encipher
ERROR Message in buffer!
root@cs519-vm:/home/user/OS-Theory-Pseudo-Character-Driver#
```

## D. Difficulties and Challenges:

1. We had difficulties in understanding when to copy something from kernel space to userspace and vice versa. We ultimately solved this problem by using copy_from_user() and copy_to_user().
2. I am currently facing a strange issue, returning specific error codes from Iocttl function from kernel space just returns “-1” to user space. Due to this I was not able to handle specific error messages in cipherctl but these can be viewed via dmesg.