Keyboard Input Security Using Virtual Keyboard

By

Amar H. Dudhait

12MCEI09

INFORMATION AND NETWORK SECURITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AHMEDABAD-382481

MAY 2014

Keyboard Input Security Using Virtual Keyboard

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in (Computer Science and Engineering)
with Specialization in Information and Network Security

By

Amar H. Dudhait

12MCEI09

INFORMATION AND NETWORK SECURITY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AHMEDABAD-382481

MAY 2014

Certificate

This is to certify that the Major Project entitled **"Keyboard Input Security Using Virtual Keyboard"** submitted by **Amar H. Dudhait (12MCEI09)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering(Information and Network Security) of Nirma University, Ahmedabad is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this Project work, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree.

Prof. Sharada Valiveti
Guide and Associate Professor,
Dept. of Computer Science and Engg.,
Institute of Technology,
Nirma University, Ahmedabad

Dr. Sanjay Garg
Professor and HoD
Dept. of Computer Science and Engg.,
Institute of Technology,
Nirma University, Ahmedabad,

Dr. Ketan Kotecha
Director
Institute of Technology
Nirma University, Ahmedabad

Undertaking for Originality of the Work

I, **Amar H. Dudhait**, Roll. No. **12MCEI09**, give undertaking that the Major Project entitled "**Keyboard Input Security Using Virtual Keyboard**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science and Engineering(Information and Network Security)** of Nirma University, Ahmedabad, is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

Signature of Student

Date:

Place:

Endorsed by

Prof. Sharada Valiveti

(Signature of Guide)

Acknowledgements

I would like to give much more thanks and feel deep gratitude to course Co-ordinator and my guide **Prof. Sharada Valiveti** for her visionary guidance, constant inspiration and constant encouragement throughout the course of this thesis. The silent pool of blessing, kind help and non-comparable guidance given by her quite often will carry me a long way in the unknown journey of my life on which I am about to go. I am really and heatily grateful for her invaluable guidance and assistance, without which the accomplishment of the task would have never been possible. I also thank her for giving me kind opportunity to explore into the real world.

I am obliged to **Dr. Sanjay Garg**, Professor and Head of Department, Computer Engineering, Nirma University, Ahmedabad, and **Dr. K. Kotecha**, Director, Institute of Technology, Nirma University, Ahamedabad, for the valuable information provided by them in their respective fields. I am grateful for their cooperation during the period of my assignment.

Lastly, I thank my **beloved parents, brothers and friends** for their steady and never-ending encouragement without which this assignment would not be possible.

-Amar H. Dudhait
12MCEI09

Abstract

Main purpose of this study is to analyze security issues related to keyboard input security and then apply security at application level. Such security is not provided by current network protection technology or server protection technology. Due to this, area of research in keyboard input security has been one of major focus recently. In order to contribute to the project, an application has been built that secures keyboard input information. Solution of earlier researches and current application trends have suggested technology for Windows environment, while some issues has been analyzed regarding keyboard input security in Linux environment. Today Linux systems usage is rapidly increasing, so the solution will be helpful to provide input security in such situation. And also virtual keyboard application has been discussed which is able to secure keyboard input information in windows environment as part of implementation.

Contents

List of Figures

Chapter 1

Introduction

Recently, there has been increase of financial frauds in which hacking tools often perform personal information stealing by passively capturing information from victim's machine and then transmitting that crucial information to hacker's website automatically. During the whole time, victim is not aware of such passive attack.Such types of frauds can not be tackeled with current network security technology or server security technology. In addition, ease of access of hacking tools makes this sensitive scenario even more sensitive for input information theft. So, the need arises to introduce new area of research to provide security to keyboard input information.In earlier researches, method that had been carried out was that development(updation) of keyboard security driver responsible for securing entire section of keyboard input information processing stages. In this research effort have been carried out to study existing keyboard driver.And also one application tool has been designed that secures keyboard input information.

1.1 Device Driver

In real world, function of driver is to manage, to control and to monitor the particular entity. In same way, piece of code that controls piece of hardware in computer system is called device drivers. So, technically, a either some hardware device or some specific core software can control most of hardware devices computer machines. Hardware device that is capable of controlling hardware device is called device controller(e.g. hard disk controllers, display controllers). Such device controller is device itself. While core software that is able to control hardware is called device driver. Device controller needs some software to run them, which is commonly called bus driver.

We can think of operating system software as divided into many layers. Among them, bus drivers lies in the bottom most layers. At such lower layers, hardware interfaces are written in bus drivers for the various corrosponding hardware protocols. Over this layer, device drivers comes into picture. These device drivers are different according to different devices and uses bus drivers internally. Due to this reason, device drivers are device specific.

Device drivers have two parts: One is device related and other is OS related. The device-related part of a device driver behaves almost same in almost all operating systems. OS-related interface of a driver, in Linux, a driver dowes further classification into three sections: network drivers, storage drivers, character drivers. We will discuss character driver in brief because keyboard driver falls into category of character device driver.

1.2 Virtual Keyboard

Technically, a virtual keyboard is nothing, but tiny application module that facilitates a user to enter keyboard inputs within its module instead of typing through keyboard. A virtual keyboard can be made compatible with multiple input devices, for example, a touch screen, an actual computer keyboard and a computer mouse. Today, most of systems contain built-in virtual keyboard software component. Bi-lingual or multi-lingual users can be benifited from such virtual keyboard. Because, such application provide them easy to swith mechanism between different character sets or alphabets. Nowadays, virtual keyboards are quite alternative for an on screen input method in most of devices with no any other general keyboard device in place. It means PDAs, tablet PCs, mobile phones are equipped with virtual keyboard as software instead of seperate hardware along with them. If security issues are considered regarding virtual keyboards, then both advantages and disadvantages for using virtual keyboards can be there. Advantage of using virtual keyboard is to reduce the risk of keylogging. On the contrary, the malware that are being used to monitor the display and mouse movement in order to get keyboard input are difficult to detect. In spite of having virtual keyboard, such malware make them ineffective to provide security. So, today most of bank uses virtual keyboard component in their websites to prevent keylogging. In this way, virtual keyboard is very useful in context of keyboard input security.

Chapter 2

Definition and Scope

2.1 Project Definition

Today most of computer systems are equipped with some applications which are responsible for recording any keystoke information. Such applications are called key-loggers. So, the need arises to have some functionality provided by application or by the system which in turn provide security for such input information theft cases. This project is dedicated to provide solution of input information theft at application level. Project is aimed to study keyboard driver software so that any unauthorized program cannot steal information from keyboard port. The project is aimed to develop an virtual keyboard appplication which provides keyboard input security in exisitng system such that attempt of access of keyboard port by hacking tools will have no effect.

2.2 Project Scope

Main focus of project is to study working of existing basic keyboard device driver. In this way, how hacking tools intercept the system can be known and information about how to block access of keyboard port can be got. And scope of the project is to develop virtual keyboard application which provides keyboard input security such that any keystroke information if logged by hacking tool cannot detect actual keystrokes. Rather than, such malicious tools can only get random characters instead of original keystrokes in their respective log files. Input information security that is provided by application is only for alphanumeric characters. Other keys like Shift,Control,Tab,BackSpace etc. are not encrypted in our application.

Chapter 3

Literature Survey

Basically, there are basic two approaches to deal with keyboard input security: Hardware approach and Software approach.

3.1 Hardware Approach

Hardware method uses another device which encrypts keyboard input information. There exist a number of encrypting devices from the simple one which shifts the inputs using shift register circuits to sophisticated one which converts the inputs with the combination of time variable using the system clock signal and the Arithmetic circuit. Such hardware methods prevent threat of leakage through message hooking method. But with this method there still exists problem at system level. But, this method cannot solve the problem of data leakage due to keyboard IO port scans. Another drawback of hardware method is that devices are difficult to modify and are expensive.[1]

3.2 Software Approaches

3.2.1 Screen Keyboard Method

One software method is screen keyboard method that is outside of encoding method which used the keyboard security driver when a hacking tool has been found at a system level and is where keyboard input data is sent through a mouse click. The input data that is clicked by a mouse is connected from the system message queue and sent. But, issues other than the system message queue generate problems where keyboard input data is leaked due to hacking tools.[2]

In above method, there exists one problem. The problem is that when hacking tool is found installed on lower level than keyboard security driver, keyboard security driver needs to be reboot and loaded on lower level than hacking tool. In this process, keyboard inputs are lost and never recovered when system is started.[3]

3.2.2 Effective Keyboard Security Method

Main Contribution of this research is that from the keyboard IO port to the interrupt vector table, the access to the keyboard input data through a hacking tool is blocked from its source and encoded keyboard input data is directly sent from security driver to Web page security control. As implementation point of view, this system is consisted of secure web page control installed in the server, the Debug exception processing installed in security driver, interrupt vector monitoring and keyboard input data encryption. Various methods to provide security to system are described as following:

Resolution of keyboard Input / Output Port Scan: Basically two methods can be used to secure keyboard input information. One is to delete information using hardware control command and other one is to control(or to block) access to keyboard port itself. However, former method cannot prevent interruption of information in keyboard port by hacking tool in process of deletion. In addition, it results in slower response because keyboard input process is performed twice. The later method, access control method which secures input by deleting register wherein keyboard input data is stored when hacking tool which tries to access keyboard I/O port, is detected using debug exception feature of operating system.[4] [5]

3.3 Keyboard Internals

3.3.1 Keyboard Basics

Keyboard device deals with two controllers: One is in keyboard itself (Keyboard Encoder) and other is on mother-board (keyboard controller). On-board keyboard controller receives data in form of bytes from the keyboard encoder. The protocol that is being used by keyboard interface determines that how such data are sent. This is often a USB connector or 6 pin Mini-DIN connector.

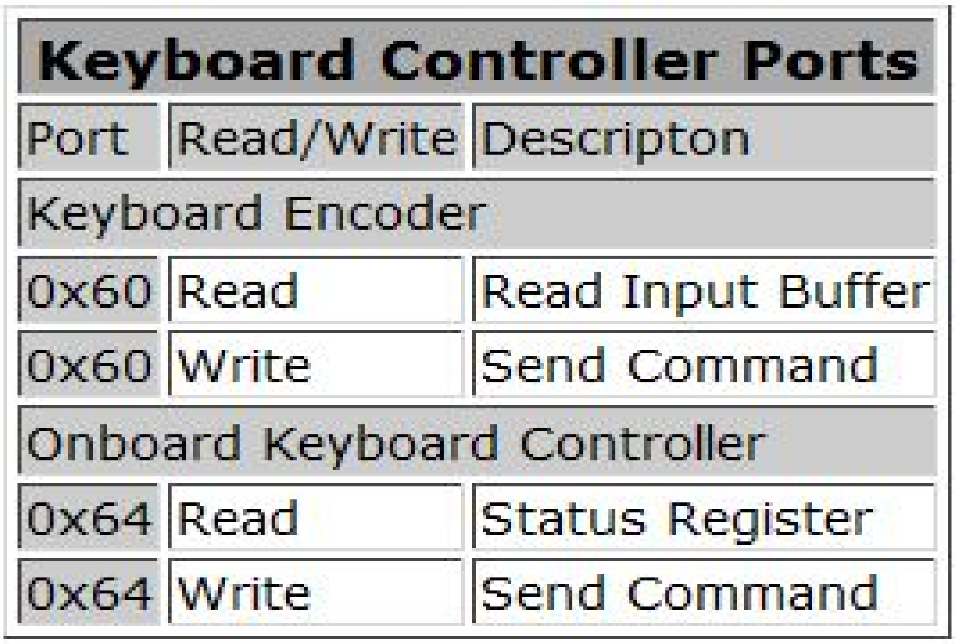
A scan code can be termed as unit of a data packet representating key state. If a key is held down, releaased or pressed, on-board keyboard controller receives relavant scan code value. There are two categories of scan codes: break codes and make codes. A break code is sent when a key is released while make code is sent when a key is held down or pressed. There is a unique make-break code pair for every present key on the keyboard.

Let's take some illustration. If you apply keystrokes like ctrl+B on input device, what will be resulting make code ? In order to better understand this, there is need to really look in event sequences that happened. First the ctrl key is pressed, then the B key is pressed. Then the B key is released followed by the ctrl key being released. We will use default scan code set here for convinience for normal keyboards. The left-most control key make code is 0x14, break code is 0xF0 and 0x14. The make code for the B key is 0x32 while the break code is 0xF0 and 0x32. So when this event sequence occurs, the following scan codes are be sent to the machine:
Key events: Ctrl Down B Down B Released Ctrl released
Scan codes: 0x14 0x32 0xF0 0x32 0xF0 0x14

Looking at the above, sequences of scan codes that are sent will be 0x14, 0x32, 0xF0, 0x32, 0xF0 and 0x14.[6]

3.3.2 Keyboard Interface

There exists mutual communication between keyboard encoder and of on-board keyboard controller. In other sense, sending a command to keyboard encoder and sending command to onboard keyboard controller is equivalent.Now, frequently used port numbers meant for keyboard will be discussed.



Keyboard Controller Ports		
Port	Read/Write	Descripton
Keyboard Encoder		
0x60	Read	Read Input Buffer
0x60	Write	Send Command
Onboard Keyboard Controller		
0x64	Read	Status Register
0x64	Write	Send Command

Figure 3.1: Port Mapping in Keyboard Interface

- Write out byte to 0x60.It is equivalent to sending a command to the keyboard encoder.
- Read in from port 0x60.It will get you data bytes from the keyboard encoder.
- Write out a value to port 0x64.It is equivalent to sending a command to the onboard keyboard controller.
- Read in from port 0x64. It will return you the status byte of the keyboard controller.[6]

3.4 Character Device Driver

If device driver for byte oriented operations are written, then it is called character device driver. As keyboard device is character device, discussion about the basic working of character device is preferred. Basic Flow:

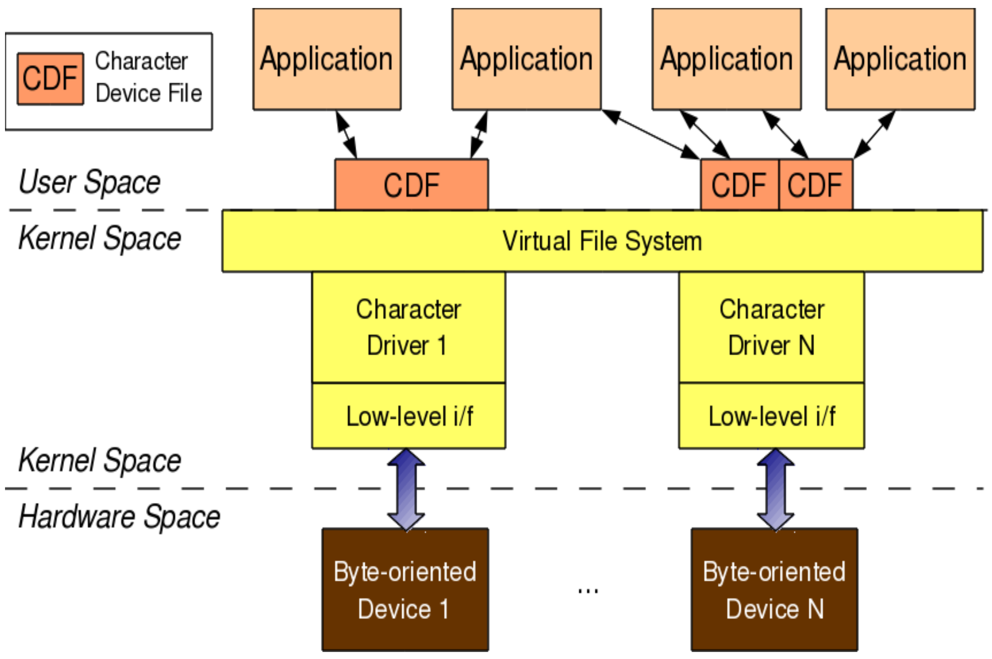


Figure 3.2: Character Device Operation

- User application performs normal file operations. Such operation is done on character device files unlike regular files.
- Operations that are carried out are not exactly same as normal file read write operations. Instead, regular file opearations are translated into corresponding functions in the relavant character driver code by Virtual File System (VFS) which plays a role as intermediary.
- Finally, lower level invocation are performed by these functions to actual device to get intended results.

In complete flow of character driver operation, main four objects are participated: (From Lower to Higher) character device(keyboard device), character device driver(keyboard driver), character device file, application. Driver operation continues only after these four are explicitly connected. Now, open system call on the device file connects an application to a device file. Second, linking of device files with device driver are done by special registration. This registration process is done by the driver. Then, linking between the driver with its device operations which are low level in nature is done. Thus, complete flow of character device opeartion is formed. [7]

3.5 Module

3.5.1 Definition

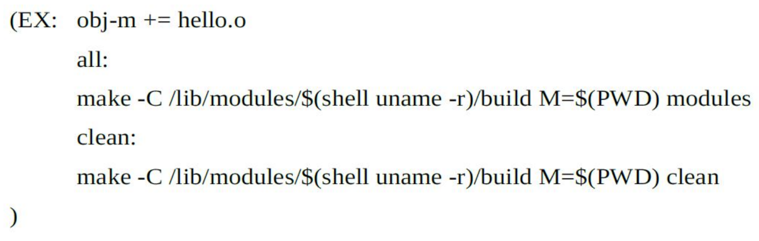
Device driver can be written in form of module. Such module then are dynamically accpteed(loaded) in kernel. Due to, loading of modules in kernel, system reboot is not necessary. It is possible to write device driver as a module that can be dynamically loaded in kernel without requiring the system to be rebooted. Device specific code of device driver is encapsulated in specific module (.ko). All pre-built modules lies under root(/) in the very standard place in every linux file system. Relative path for such modules (under kernel source tree structure) can be described as /lib/modules/kernel-version/kernel. [8] [9]
Commands: (must run with root privileges)

- insmod *module-name* loads dynamically specific module file(.ko) in kernel.
- lsmod lists currently loaded modules.
- rmmod *module-name* unload any modules
- dmesg write kernel messages to standard output.

How to Run Modules

It is total seven step process:

- Make one module file with .c extension
- Make one Makefile



```
(EX:  obj-m += hello.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
)
```

Figure 3.3: Make File

- *make -f Makefile* (It will generate .ko , .o files automatically)
- Load the Module - *insmod module-name*
- To display kernel message to standard output dmesg
- Unload Module *rmmod module-name*

3.6 Application Study

There are many applications available whose main function is to protect keyboard input information. Some of example application names are KeyScrambler, Oxynger KeyShield etc.

3.6.1 KeyScrambler

KeyScrambler software has inherent preventive mechanism in order to guard the user's confidential information by means of encrypting input information entered by user in real time. KeyScrambler has been ranked for the top to deal with almost any types of keylogger softwares. In addition, it also provides best security for the machines that acts as zombies. KeyScrambler's need in order to operate in the system is around a quite little and provides user transperancy. Comapatible applications that are supported by keyscrambler are 30 browsers and 150+ programs to prevent keylogging. KeyScrambler can also deal with smartphone applications. Because Keyscrambler has ability to deal with any type of unknown malicious keylogging programs and softwares, it hasbeen considered as best software for input information security. One of the eye-catching feature of keyscrambler software is that it keeps you showing characters that you have typed in encrypted form.Now encryption technology adopted by KeyScrambler will be discussed.[10]

- It applies symmetric cryptography and asymmetric cryptography priciples both.
- Blowfish(128-bit) algorithm for single key encryption and RSA(1024-bit) algorithm for public key encryption is used.

How It Works?

- When you type character in editor or in browser or in any compatible application, KeyScrambler performs encoding your input in unreadable manner in real time at the operating system level. Since KeyScrambler is being executed from kernel level, compromising KeyScrambler's cryptic operation is very challenging one.
- Since encrypted input goes through system, it remains undetecteable and non-readable throughout. Because it remains encrypted in line, any captured by a keylogger sitting on your computer, or whether the keylogger is known or brand new.
- KeyScrambler begins to decrypt encrypted input that are just received by destination application. Finally, you get original key inputs in your application.

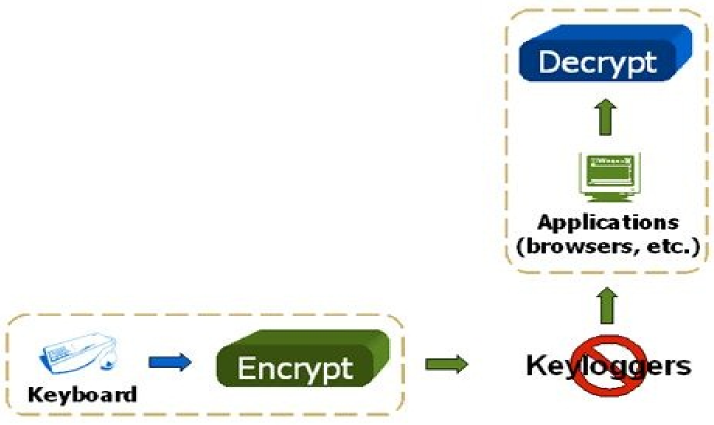


Figure 3.4: KeyScrambler: How It Works

3.6.2 Oxygner KeyShield

Another alternative for KeyScrambler is Oxynger KeyShield. This software is basically virtual keyboard with security mechanism in itself. Thus, it helps to protect crucial user input information that are typed through keyboard such as passwords, credit card numbers etc. or logged by keyloggers. Portable version is available for this software. It is considered the best choice for input security at public computer such as in bank or in shopping mall. The basic difference between ordinary anti-keyloggers and this one is that ordinary ones guards user input against software keyloggers while Oxygner guards user input against both hardware keyloggers and software keyloggers. Oxygner uses some specific method to send user input to intending application. It uses secured data channel in order to send user inputs to intended application.[11]

Chapter 4

Proposed Approach

4.1 Current Scenario

Right now, many desktop user systems or servers located in global internet network runs in Windows Environment because of its popularity. So, there may be large possibility of occurences of fraud cases like input information theft in internet. A kind of attack in which keyboard inputs are being captured passively is difficult to be resolved by existing server protection technology or network protection technology. So, many successful attempts has been done to provide keyboard(P/S2) input security in windows environment. By the hardware method, keyboard input information can be encrypted to avoid exposure of confidential information to untrusted user by hardware device. There exists many anti-keylogger software which needs to be installed and needs to be configured in order to protect keyboard input information. They don't come system inbuilt, rather installation for the same is needed. Eariler two well-known keylogger applications was discussed: Keyscrambler and Oxygner. Now, pros and cons of such application will be discussed. KeyScrambler provides on-fly encryption of user input information. But, it does not come as portable application. Nowadays, almost people perform their confidential transactions on public places where revelation of confidential information is quite often due to non-trusted network or computers.Such networks may not use such antikeylogging softwares or may use deliberately keylogging softwares. Such problems has been overcome by softwares such as Oxygner Keyshield. This software comes as portable. If such software is installed at public places, then there is no need to worry about such input information thefts. But other problem arises in case of Oxygner Keyshield and like ones that they all are system dependent. They are compatible with almost all kinds of windows operating systems. But they are quite unuseful for Linux systems and other non-Windows Systems.

4.2 Proposed Scenario

Today usage of linux systems are rapidly increasing. In spite of the fact that linux system provides good security, there is chance of keyboard input information theft by some malicious applications. So, the project is aimed to study keyboard driver code in order to understand security security provided by it. The keyboard type for which driver is to be studied is P/S2. And second, application have been developed called virtual keyboard which inherently provides security to keyboard input information. When any user types through virtual keyboard, key events that are being logged by keylogger software (if exists in system) have no effect. Because as soon as key strokes are being logged, they are encrypted by vitual keyboard application internally.So, even if keylogger application want to log keystrokes, only they can receive information is encrypted keystrokes instead of original keystrokes given by user. Keyboard input security that is provided is for alphanumeric characters. Any other inputs like Modifier Keys(Tab,CapsLock,Shift,Ctrl,Alt), Function Keys(F1-F12) and other keys(Inser,Home,Up,Down etc...) are not encrypted by application. Other applications such as KeyScrambler and Oxygner KeyShield have been discussed. One basic advantage of application is that it is designed in Java Platform. It is known that Java applications can run on any system with any platform having JVM installed. So, designed application is portable plus can be run on any machine having JVM. In this way, our virtual keybaord application is helpful to deal with keyboard input security.

Chapter 5

Device Driver Implementation

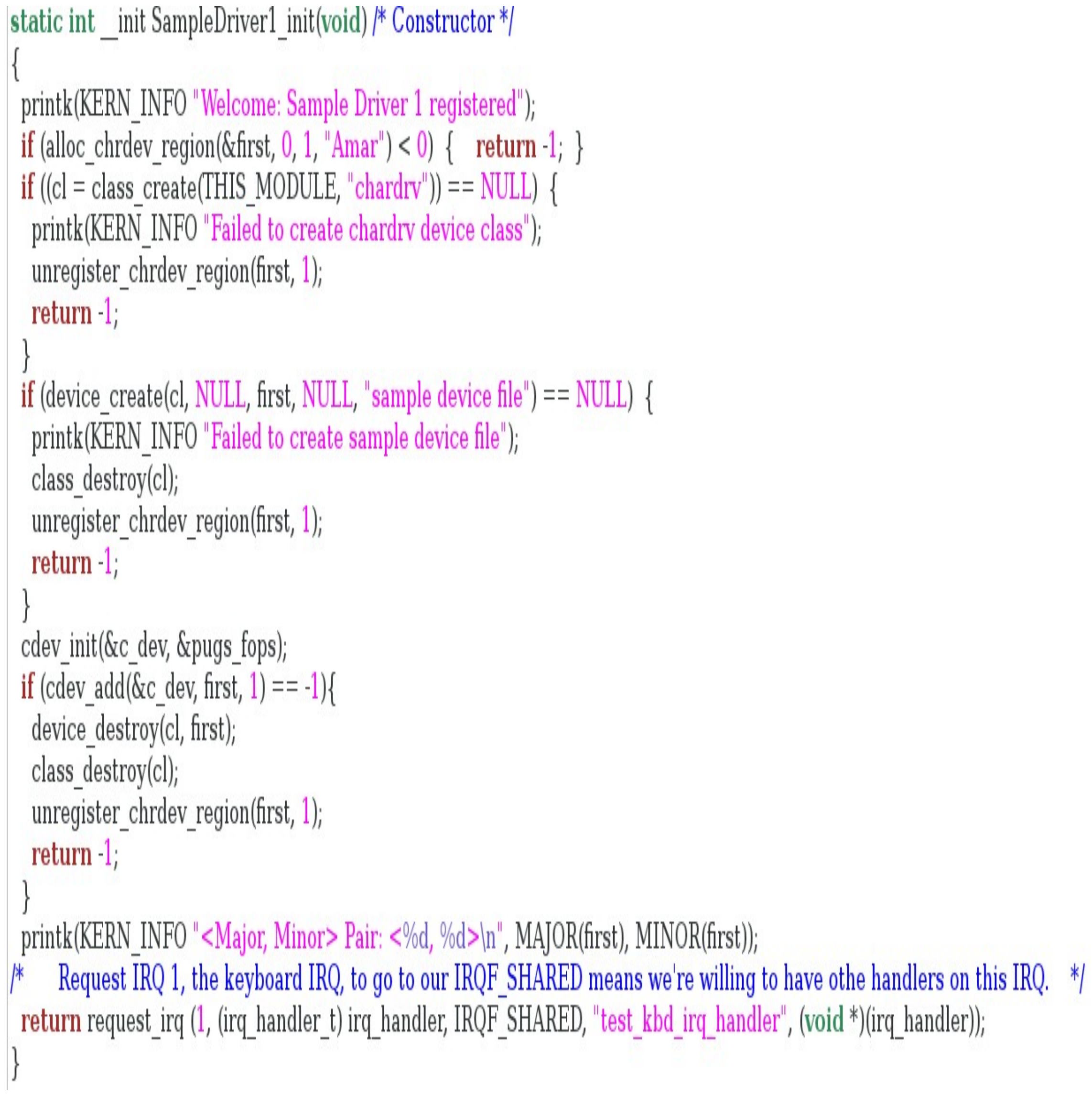
5.1 Sample Device Driver

The System that is used for experiment is Linux Fedora 16 and P/S2 keyboard. In order to understand driver code, First there is need to know basic format of any device driver code. So, sampe code of one device driver is mentioned whose objective is to notify pressing [Esc] key event to the console by printing message.

Explanation: When any module is inserted into running kernel,its init function is called.Init function contains initialization code plus any other task that you want to perform with module. And when module is removed from linux kernel,it calles exit function which finally unregisters or destroys any software resources that have been used by modules. In Init Module, Device file along with (Major,Minor) pair is needed.Connecting the device file with the device driver involves three steps:

- Registering for the Major,Minor range of device files.This is done by register.chrdev.region() function.
- Dynamicaly create device file under .dev directory, first we need to create device class (class.create() function) and then device file (device.create() function).
- Linking the device file operations to the device driver functions.

Virtual File System (VFS) decodes the file type and transfers the file operations to the appropriate channel, like corresponding device driver. Now, for VFS to pass

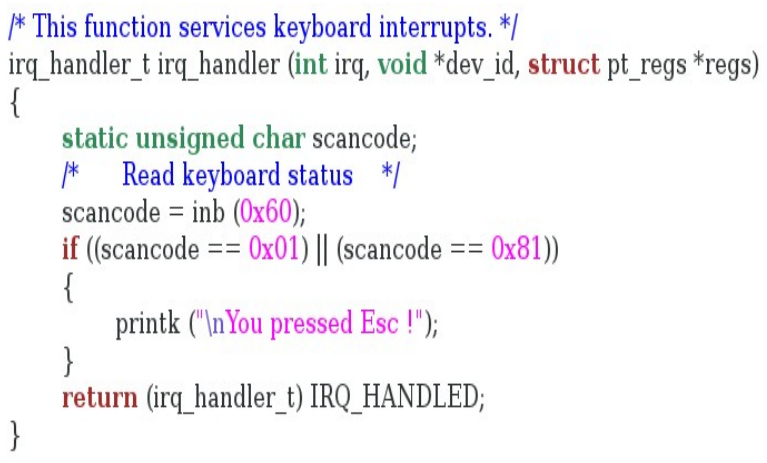


```
static int __init SampleDriver1_init(void) /* Constructor */
{
  printk(KERN_INFO "Welcome: Sample Driver 1 registered");
  if (alloc_chrdev_region(&first, 0, 1, "Amar") < 0)  {   return -1;  }
  if ((cl = class_create(THIS_MODULE, "chardrv")) == NULL)  {
    printk(KERN_INFO "Failed to create chardrv device class");
    unregister_chrdev_region(first, 1);
    return -1;
  }
  if (device_create(cl, NULL, first, NULL, "sample device file") == NULL)  {
    printk(KERN_INFO "Failed to create sample device file");
    class_destroy(cl);
    unregister_chrdev_region(first, 1);
    return -1;
  }
  cdev_init(&c_dev, &pugs_fops);
  if (cdev_add(&c_dev, first, 1) == -1){
    device_destroy(cl, first);
    class_destroy(cl);
    unregister_chrdev_region(first, 1);
    return -1;
  }
  printk(KERN_INFO "<Major, Minor> Pair: <%d, %d>\n", MAJOR(first), MINOR(first));
/*     Request IRQ 1, the keyboard IRQ, to go to our IRQF_SHARED means we're willing to have othe handlers on this IRQ.    */
  return request_irq (1, (irq_handler_t) irq_handler, IRQF_SHARED, "test_kbd_irq_handler", (void *)(irq_handler));
}
```

Figure 5.1: Init Module Code

the device file operations onto the driver, it should have been informed about it. That is what is called linking the file operations by the driver with the VFS. This involves two sub-steps. 3a. Fill in file operation structure with desired file operation. 3b. Initialize the character device structure(struct cdev c.dev) with cdev.init() function. Finally, above structure is passed to the VFS using cdev.add() function.This is how init function is designed.

To be get notified by key event, interrupt handler is defined. Then, handler in init function is called at an end to passively receive keyboard device interrupts. Handler



```
/* This function services keyboard interrupts. */
irq_handler_t irq_handler (int irq, void *dev_id, struct pt_regs *regs)
{
    static unsigned char scancode;
    /*    Read keyboard status    */
    scancode = inb (0x60);
    if ((scancode == 0x01) || (scancode == 0x81))
    {
        printk ("\nYou pressed Esc !");
    }
    return (irq_handler_t) IRQ_HANDLED;
}
```

Figure 5.2: IRQ Handler Code

works like this: it continuously scans keyboard IO port(60h) and checks scan code for [Esc] key or not. If yes, it prints message at console.

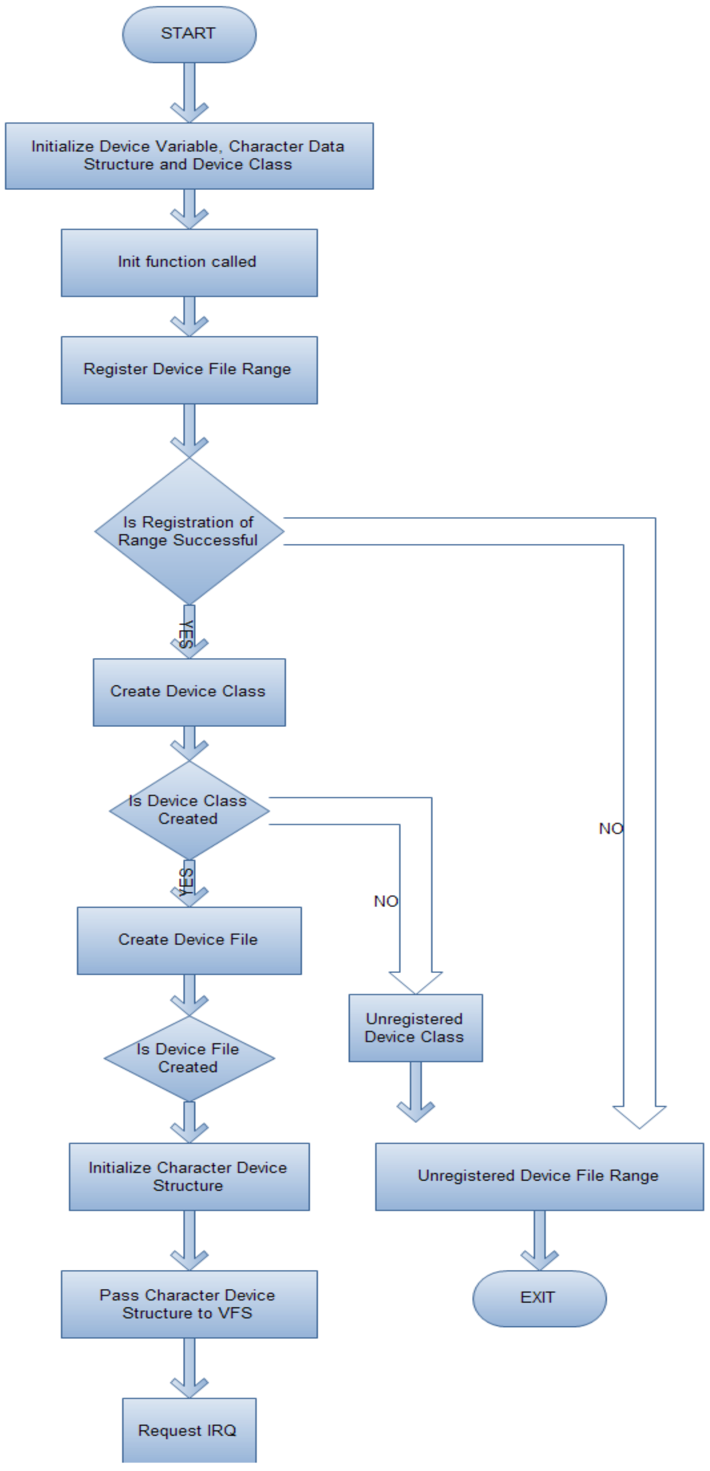

START
Initialize Device Variable, Character Data Structure and Device Class
Init function called
Register Device File Range
Is Registration of Range Successful
YES
NO
Create Device Class
Is Device Class Created
YES
NO
Create Device File
Is Device File Created
Unregistered Device Class
Initialize Character Device Structure
Unregistered Device File Range
Pass Character Device Structure to VFS
EXIT
Request IRQ

Figure 5.3: Init Module

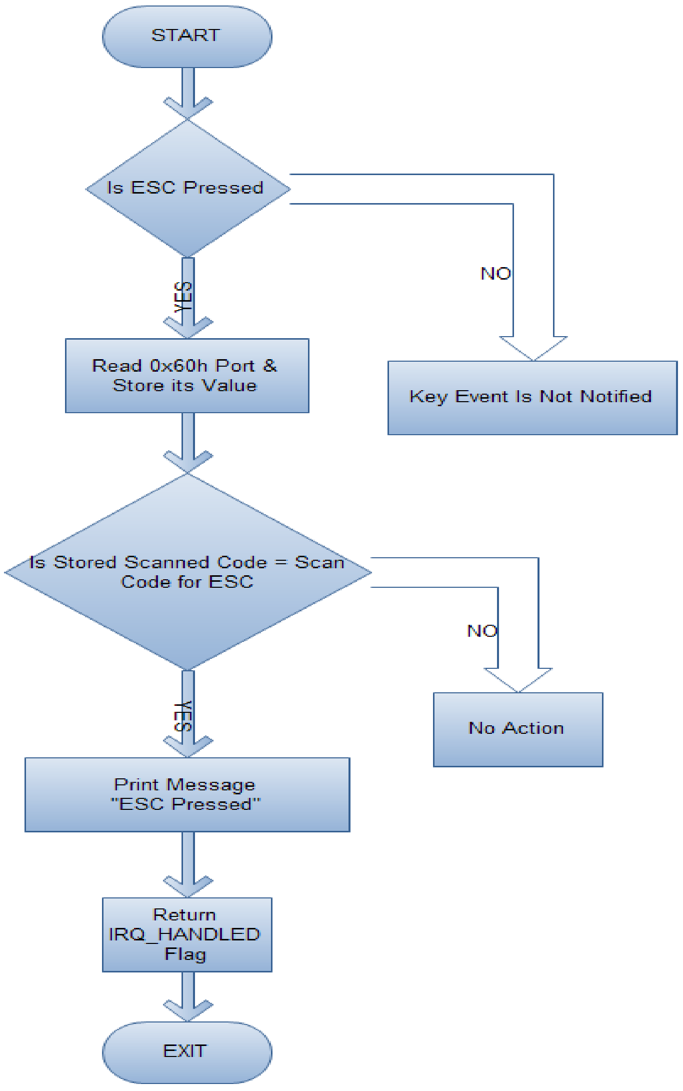

START
Is ESC Pressed
NO
YES
Read 0x60h Port & Store its Value
Key Event Is Not Notified
Is Stored Scanned Code = Scan Code for ESC
NO
YES
No Action
Print Message "ESC Pressed"
Return IRQ_HANDLED Flag
EXIT

Figure 5.4: Interrupt Handler

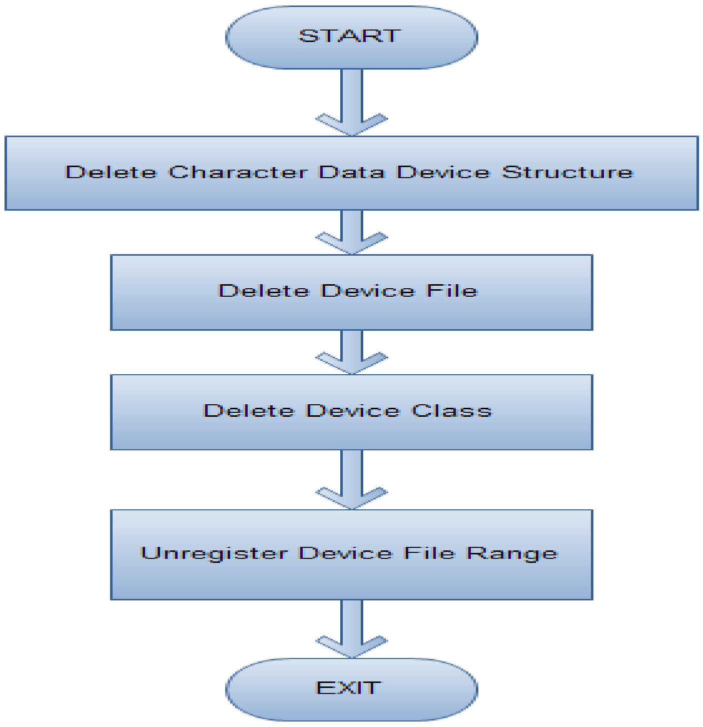

START
Delete Character Data Device Structure
Delete Device File
Delete Device Class
Unregister Device File Range
EXIT

Figure 5.5: Exit Module

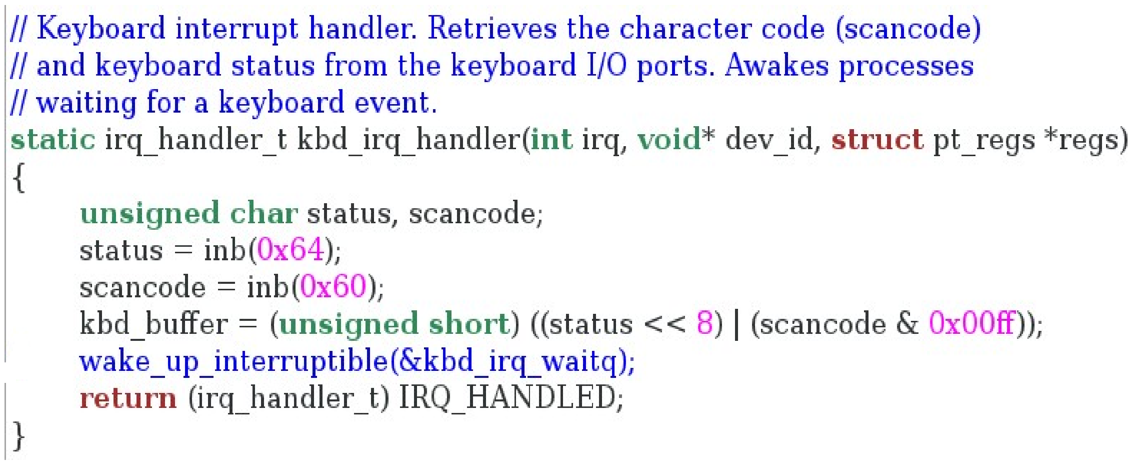
Chapter 6

Keyboard Device Driver Implementation

6.1 Basic Keyboard Device Driver

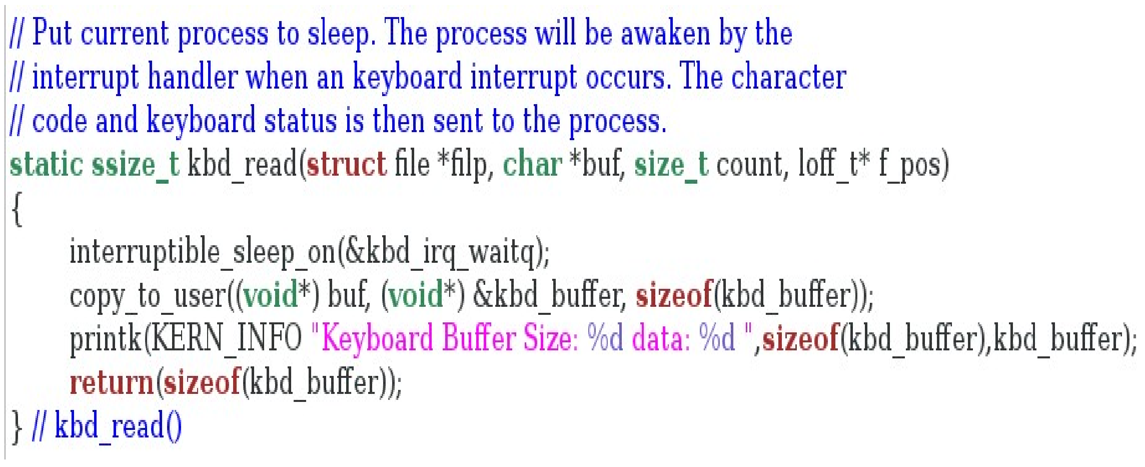
As implementation part, sample device driver is designed. Then other simple keyboard device driver is designed whose function is to pass keyboard scan code and status register information to user process which issued system call for keyboard read. Function is only designed for reading keystrokes, not for writing user-typed keystrokes recognized by system.

So, we will see how this code works: This driver code listens for keyboard interrupts on IRQ1 and notifies processes when an interrupt is raised. So when application issues any keystrokes, keyboard interrupt is generated. So, the process that receives keystroke events will run(or listen for keystrokes) in background until an keyboard interrupt occurs.When key is pressed, process will be put on sleep and driver's interrupt handler is called. Now the driver's interrupt handler, will awake all of process sleeping in waiting queue at once.Then handler receives scan code and keyboard status from the keyboard via 60h and 64h ports respectively.



```
// Keyboard interrupt handler. Retrieves the character code (scancode)
// and keyboard status from the keyboard I/O ports. Awakes processes
// waiting for a keyboard event.
static irq_handler_t kbd_irq_handler(int irq, void* dev_id, struct pt_regs *regs)
{
        unsigned char status, scancode;
        status = inb(0x64);
        scancode = inb(0x60);
        kbd_buffer = (unsigned short) ((status << 8) | (scancode & 0x00ff));
        wake_up_interruptible(&kbd_irq_waitq);
        return (irq_handler_t) IRQ_HANDLED;
}
```

Figure 6.1: Sample 2 IRQ Handler



```
// Put current process to sleep. The process will be awaken by the
// interrupt handler when an keyboard interrupt occurs. The character
// code and keyboard status is then sent to the process.
static ssize_t kbd_read(struct file *filp, char *buf, size_t count, loff_t* f_pos)
{
        interruptible_sleep_on(&kbd_irq_waitq);
        copy_to_user((void*) buf, (void*) &kbd_buffer, sizeof(kbd_buffer));
        printk(KERN_INFO "Keyboard Buffer Size: %d data: %d ",sizeof(kbd_buffer),kbd_buffer);
        return(sizeof(kbd_buffer));
} // kbd_read()
```

Figure 6.2: KBD Read Function

Chapter 7

Virtual Keyboard Application

7.1 Overview

An application that mentioned here is designed in Java. So, this application can be run by any machine with any operating system (Windows or various Linux Distro.) which have JVM installed in.
Most of hacking tools like keyloggers use "GetAsyncKeyState" WinAPI function internally to get keystrokes from keyboard. Here, GetAsyncKeyState() function determines whether a key is up or down at the time the function is called and sets most(higher) significant bit to one when particular key is pressed and also returns pressed key's ascii code. So our application calls similar WinAPI function "GetAsyncKeyState" through another program called antikeylogger. When user types through virtual keyboard,then it notifies key pressed event through most significant bit set(by WinAPI function). In addition, when "GetAsyncKeyState" function return ascii code for pressed key,our antikeylogger program garbles ascii code for pressed key. By performing garbling operation,any hacking tool if installed in machine only gets random keystrokes information.In this way, our virtual keyboard application is able to secure keyboard input information.

7.2 How It Works?

When virtual keyboard application is launched, then first, it asks the user that for which built-in application(e.g. word,notepad,firefox etc.) he want to apply virtual keyboard feature. Based on the selection of checkboxes given, virtual keyboard application pops up virtual keyboard when selected application is started or running.When selected application is closed, then virtual keyboard also disappears from the screen.

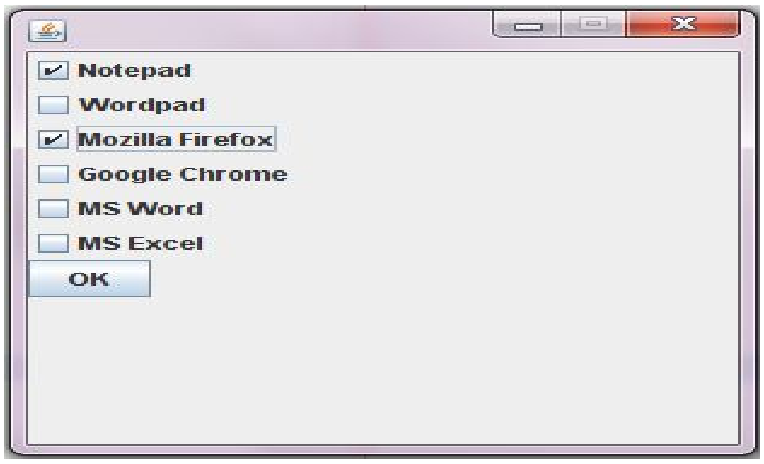

[✓ Notepad]
Wordpad
[✓ Mozilla Firefox]
Google Chrome
MS Word
MS Excel

Figure 7.1: VKeyboard Option Window

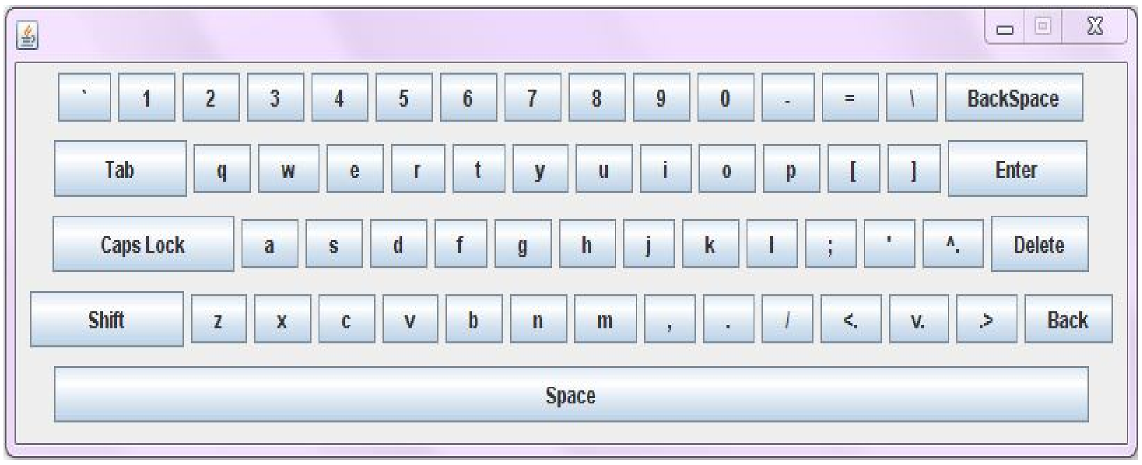
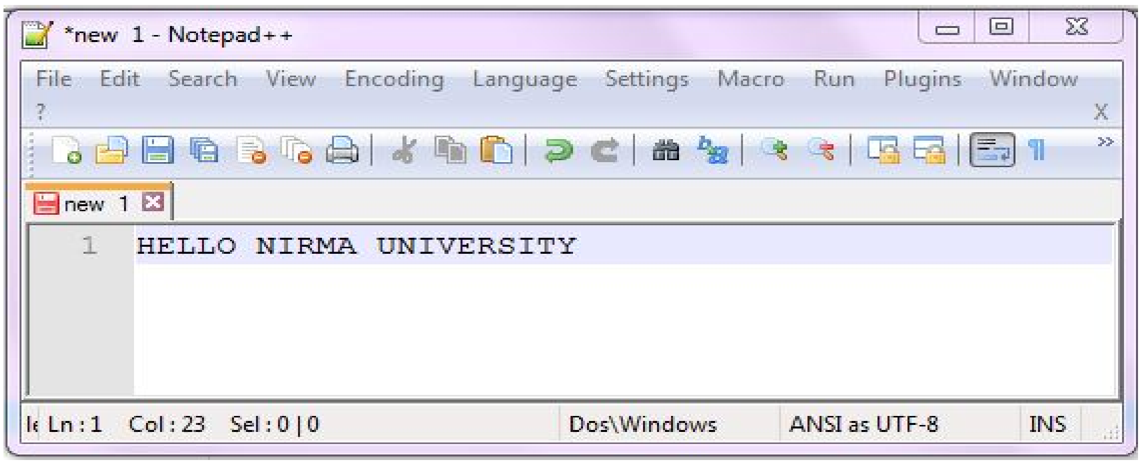

` 1 2 3 4 5 6 7 8 9 0 - = \ BackSpace
Tab q w e r t y u i o p [ ] Enter
Caps Lock a s d f g h j k l ; ' ^. Delete
Shift z x c v b n m , . / <. v. .> Back
Space

Figure 7.2: Virtual Keyboard Main Window

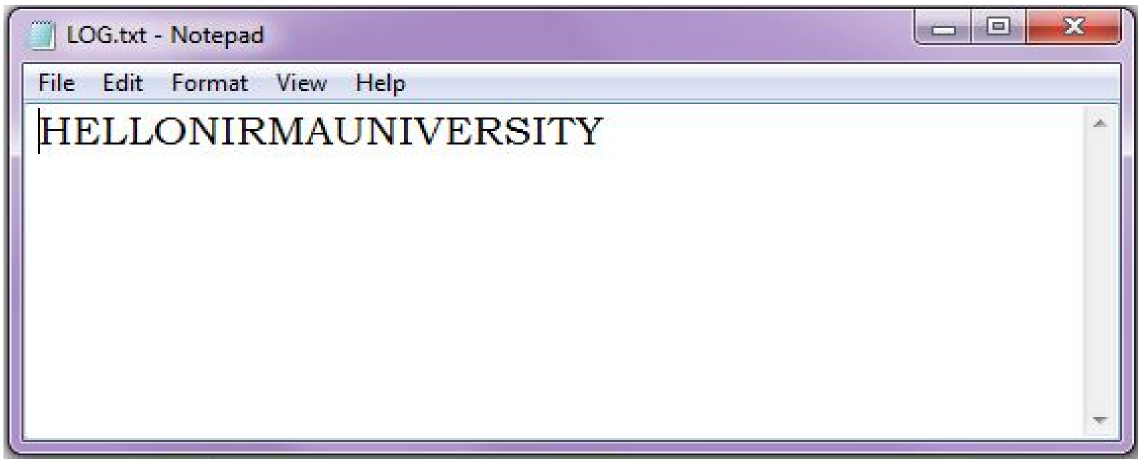
- Using virtual keyboard, any text is entered. If there is any keylogger, then the keystroke stored by it are stored in some log file.


*new 1 - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
new 1

```
1  HELLO NIRMA UNIVERSITY
```

Ln : 1 Col : 23 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

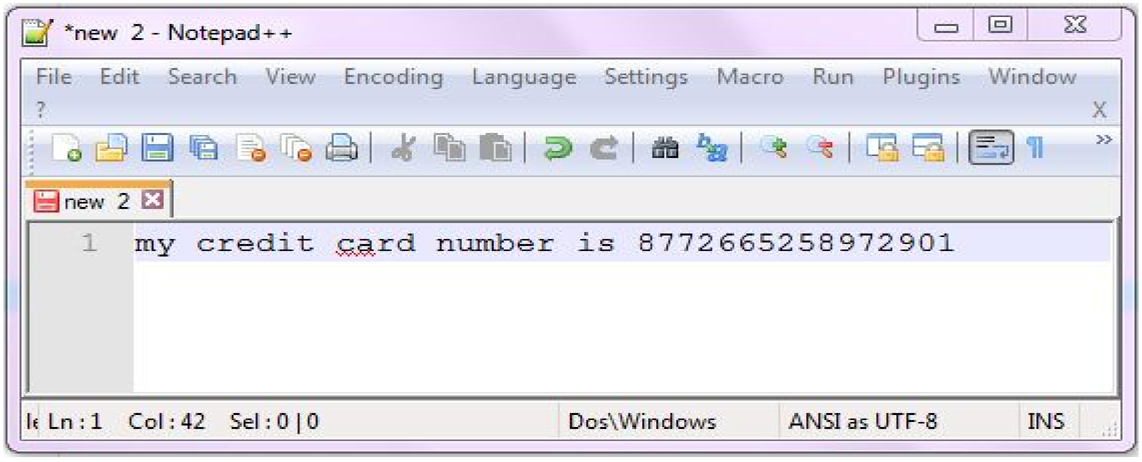
Figure 7.3: SampleText Input1



```
HELLONIRMAUNIVERSITY
```

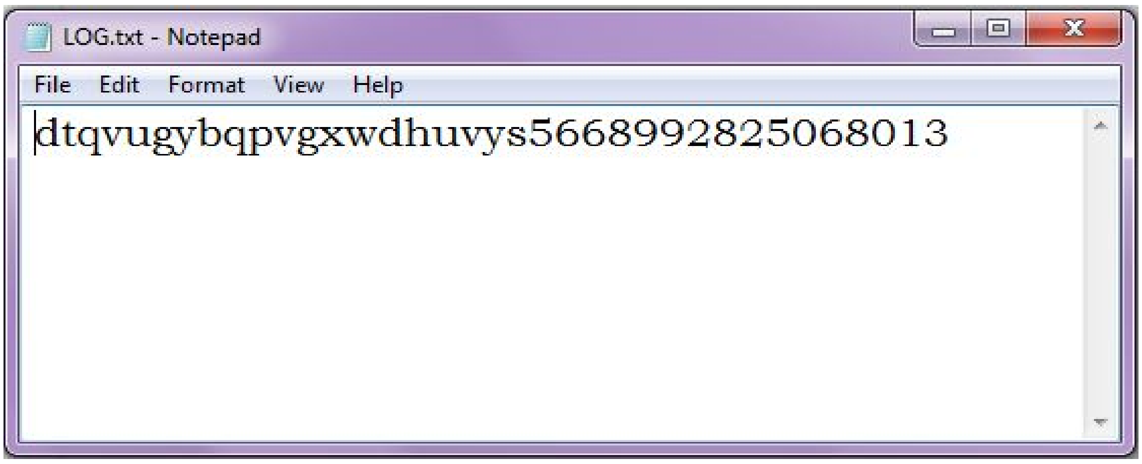
Figure 7.4: Sample Output1

- But since antikeylogger program is running, thus typed keystrokes will get garbled. So, any actual keyboard input information is not available to any hacking tools.


*new 2 - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
new 2
Ln : 1 Col : 42 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

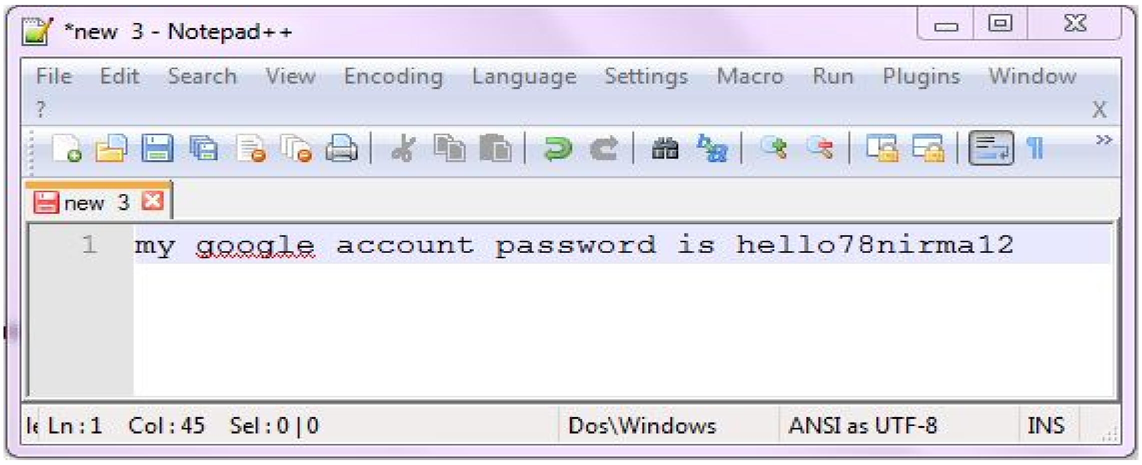
```
my credit card number is 8772665258972901
```

Figure 7.5: Sample Input2



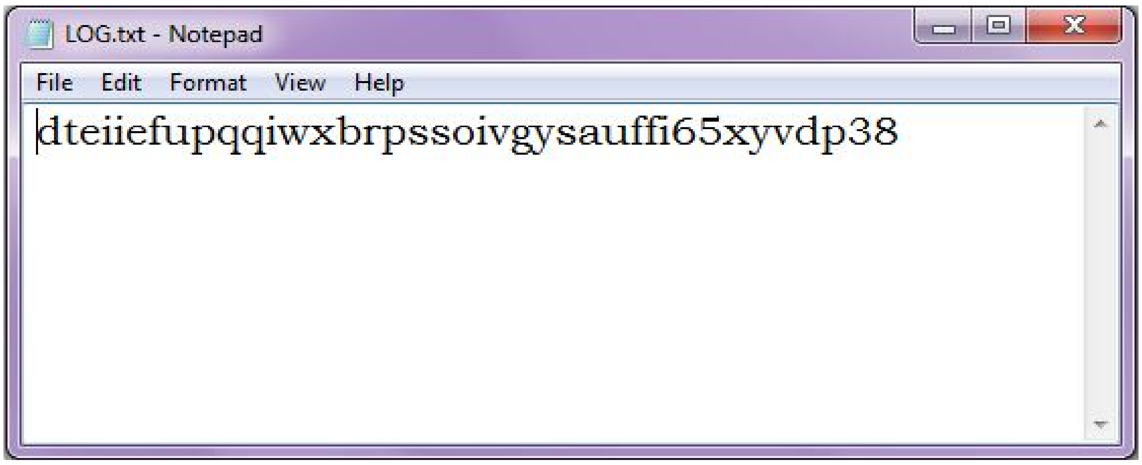
```
dtqvugybqpvgxwdhuvys5668992825068013
```

Figure 7.6: Sample Output2


*new 3 - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
new 3
Ln : 1 Col : 45 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

```
my google account password is hello78nirma12
```

Figure 7.7: Sample Input3



```
dteiiefupqqiwxbrpssoivgysauffi65xyvdp38
```

Figure 7.8: Sample Output3

Chapter 8

Conclusion and Future Work

It is feasible to study simple keyboard device driver. And also Java Virtual Keyboard application is demostrated that is able to prevent actual character log of keystrokes that is performed by hacking tool like keylogger. In future, another feature called anti-screen logging to this application will be provided. Also other feature of mouse keylogging will be embedded into application that is responsible for protecting from mouse keylogging that is done by some keyloggers nowadays.

References

[1] Michael F. Angelo *Method and Apparatus For Providing Secure and Private Keyboard Communications in Computer Systems* US Patent, 5748888, 1998.

[2] Ahn Lab Inc. *Keyboard Security Method* Korea Patent, 10-0496462, 2003.

[3] Chang, Hangbae and Lee, Sijin and Eum, Wonsub *The Research of Keyboard Security for u Trading Book: Intelligent Pervasive Computing(IPC), 2007 , pages 165-169.*

[4] Shakshuki Elhadi, Luo Zhonghai, Gong Jing. *An Agent-Based Approach to Security Service Journal of Network & Computer Applications, Vol. 28, Issue 3, 2005.*

[5] Chang, Hangbae and Kim, Kyung-Kyu and Lee, Hosin and Kim, Jungduk *The Design and Development of a Secure Keystroke System for u Business Book: Computational Science and Its Applications - ICCSA 2006 , pages 255-261.*

[6] Mike *Operating Systems Development - Keyboard* `http://www.brokenthorn.com/Resources/OSDev19.html` *2009*

[7] AnilKumar Pugalia *Linux Character Devie Drivers.* `http://www.linuxforu.com/2011/02/linux-character-drivers/` *, Apr 1, 2011*

[8] Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman *Book: Linux Device Drivers, 3rd Edition*

[9] Sreekrishnan Venkateswaran *Book: Essential Linux Device Drivers , Mar 27, 2008*

[10] KeyScrambler *Most Advanced Anti-keylogging Solution* `http://www.qfxsoftware.com/`

[11] Oxygner *Ultimate Protection from passwords by Hacking From Keyloggers* `http://www.oxynger.com/`