ENHANCING FILE AND SYSTEM SECURITY BY ADDING LOADABLE KERNEL MODULE FOR REMOVABLE STORAGE DEVICE

Deepak Dilip Mahajan [1], Prof. A. B. Chougule [2]
[1]*Computer Science & Engg. Department, KIT's College of Engineering, Kolhapur, Maharashtra, (India)*
[2]*Computer Science & Engg. Department, BVP's College of Engineering, Kolhapur Maharashtra, (India)*

ABSTRACT

As computer systems are secured using password mechanism, attempts are made to take undue-advantage by hacking passwords. Breaking password of the system pose a great threat and proves to be a major handicap in file security. Ease with which login step can be bypassed, is point to be concerned. Generating evidence against intruder is another challenge in digital forensic investigation. Application level solutions regarding this issue are frail. Use of biometric password is preferable but costly option. This invokes the need of system level security mechanism with low cost hardware. This paper proposes one solution to enhance file and system security. Along with password, removable storage device can be used as additional parameter, for authenticating access to file and system. To achieve proposed technique, preinstalled plug-ins in the form of Loadable Kernel Modules (LKM) are added in Linux (kernel 2.6.x) based operating system.

Keywords - Authentication Key, Biometric Passwords, Evidence Generation, Loadable Kernel Modules (LKM), Password Security Threats, Removable Storage Device, Udev Rules.

I. INTRODUCTION

Operating systems maintain user account, to provide logically separate user-space for each user. User's account is secured by login mechanism. Only user with valid username and password is allowed to login. When a file is created, by default the logged user is considered as an owner of that file. Owner of the file can provide access rights to other users, for accessing the file. User with administrative privilege can assign privileges to other users, for accessing files as well as services provided by the system. Files are protected by access control mechanism, which strongly relies on password. Hence to secure file, securing password is mandatory.

The main threat is, intruder or hacker can break or hack the password by brute force attack or with the help of key generator software [11]. One can even bypass login step, by using software tools [12]. Once intruder is passed through login, access control mechanism does not restrict intruder from accessing files, because of successful login. Application level solutions like locking or encrypting the file are available to secure the file, but intruder with administrator privileges can remove locks. Keeping file contents in encrypted format involves time and space complexity. This is motivation behind providing stronger security by adding security mechanism at kernel level. This paper propose the technique to restrict intruder from accessing files, and also generate evidence against spurious user.

II. RELATED WORK

2.1 Enhancing Forensic Capabilities

Mridul Sankar Barik et al. [1] presented a solution for obtaining authentic date and time stamps of digital documents. The implemented solution uses preinstalled plug-in in the form of Loadable Kernel Module (LKM). LKM is advantageous in the sense that, it provides flexibility as this mechanism can be inserted and removed as per will of the administrator and it doesn't require any change in existing installations. Also the LKM hiding techniques can be used to further enhance the security of the proposed solution itself.

2.2 Preserving MAC DTS

Suvrojit Das et al. [2] proposed that preservation of MAC DTS can be useful for generating kernel level audit data, which can be an important parameter in host-based intrusion detection system. MAC DTS auditing mechanism can be used for collecting information regarding the activities of users and applications on the file system. MAC DTS preservation mechanism is both tamper resistant and non-by passable. It is possible to model correct usage pattern of users and applications regarding file system operations which conform to the current security policy of the operating system.

2.3 Adding Loadable Kernel Module

Md. Sarfaraj Alam Ansari et al. [3] enhanced idea of [2] by proposing kernel level Virtual File System (VFS) logger for intrusion detection. Interaction of different objects in VFS is presented in this paper. Author modified the default flow of system call execution by trapping of system call.

2.4 Using udev Rules to Control Device Related Actions

When device is attached and removed from Linux based systems, predefined actions are performed. These device specific actions are defined in /etc/udev/rules.d folder as rules. It is possible to add local rules or to override already existing, to get control over device related events. [9] Using udev rules, system can launch a script when a device node is created or deleted. Rule file's name must start with number and should have extension .rule. e. g. 71-local.rule with following code-

ACTION== "add", BUS=="usb",

KERNEL== "0-7", ATTR{serial}== "0000000000000C91B2CF5D05", RUN+="myrule.sh"

When USB bus identifies that the device with specific serial number is attached to the system, kernel run the program mentioned in rule.

III. PROPOSED WORK

In order to enhance the security, proposed work seeks to provide a software solution with hardware support. Unique serial number of removable storage device will be registered with specific user account, and can be used as a mandatory parameter for authenticating access to the system, file or specific service of that user account. Using udev rules, system can easily distinguish registered device from others. Hence system can be made to behave different when registered device is plugged or unplugged. Modifications are proposed in creat and open system calls, to provide security in file related operations. LKM is implemented to restrict intruder from accessing files, system and services, also to generate strong evidence against him.

IV. DESIGN AND IMPLEMENTATION

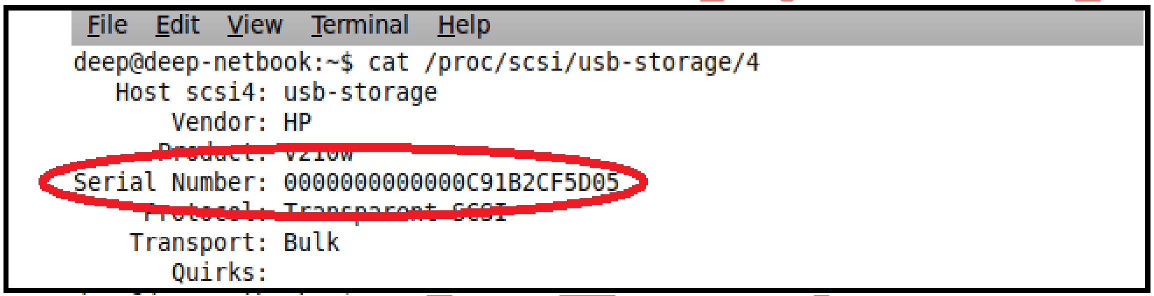
To accomplish proposed work five modules are developed.

4.1 Module 1: Device Registration

The removable storage device must be registered, in order to use it as an authentication key.

4.1.1 Finding out Serial Number

As shown in figure 1, this module identifies unique serial number of device, using device entry in /proc/scsi/usb-storage/# where # is system dependent. Use of 'udevadm' is another approach to get detailed information about any attached device.



```
deep@deep-netbook:~$ cat /proc/scsi/usb-storage/4
   Host scsi4: usb-storage
       Vendor: HP
      Product: v210w
Serial Number: 0000000000000C91B2CF5D05
     Protocol: Transparent SCSI
    Transport: Bulk
       Quirks:
```

Figure 1: Unique Serial Number of Removable Storage Device

4.1.2 Registration Entry

Retrieved serial number is registered with logged user, and entry is made in the device registration file. Structure of the device registration file is shown in figure 2. This file is also used for recovery purpose. The recovery key generated by Recovery module (Module:5) is mapped with unique device id.

User	Device id	Recovery Key
deepak	0000000000000C91B2CF5D05	9e107d9d372bb6826bd81d3542a419d6

Figure 2: Structure of Device Registration File.

4.2 Module 2: LKM for Modified Creat

Initially, when user gives command for file creation, "creat" system call is passed to kernel. In kernel area, interrupt handler calls related standard routine for file creation, and then the file is created.

The proposed LKM – secure_creat, which modifies execution flow of creat () system call, compared to regular file creation. When executed, secure_creat makes an entry of filename into "log file" associated with registered device and then gives call to original creat system call. User is allowed to execute secure_creat only when the registered device is attached to the system. Figure 3 shows summarized flow of secure_creat.

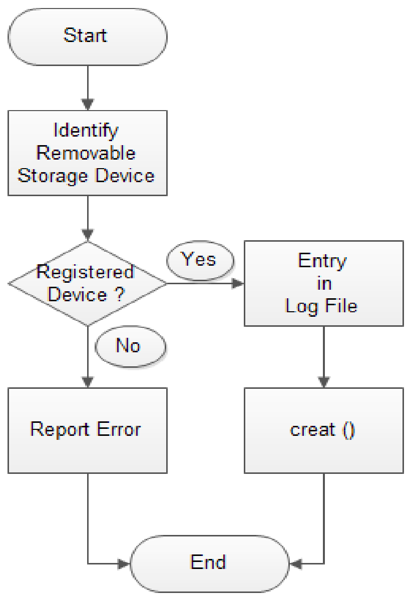


Figure 3: Flow of secure_creat

Files created using secure_creat should be allowed to access using same registered device only. So open() system call is modified and implemented in secure_open as module 3.

4.3 Module 3: LKM for Modified Open

In order to open particular file, user passes file name and the mode in which file is to be opened. System verifies privilege of user and access rights provided for the file. If verification is cleared then the file is made available to user. This scenario fails when intruder successfully logs-in.

Module 3 modifies execution flow of open system call, to provide one more verification level, every time the request for opening the file is redirected to verification through "log file entry". User, who wants to open the file, must have the same removable storage device, which was used at the time of creation of that file. Figure 4 shows flow of secure_open.

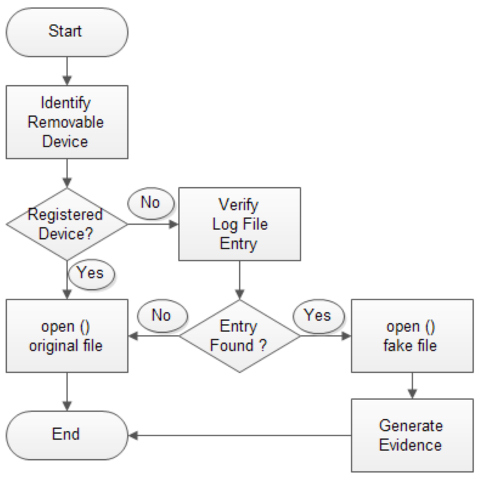


Figure 4: Flow of secure_open System Call

4.4 Module 4: Trap the Intruder

To generate evidence against intruder, module 4 sets the system to start in text mode by default. When intruder logs in, by hacking valid username and password, is not having registered device. After successful login, if module failed to found registered device, instead of default shell (bash, sh), 'fake_shell' will be made available to serve the user. 'fake_shell' is shell that is implemented to trap the intruder. 'fake_shell' provides similar interface like regular shell, so intruder is unable to recognize the trap. 'fake_shell' allows intruder to type command, but not responding as it should be, instead it activates webcam of system and take a snap of intruder. This snap and its Modification Access and Creation (MAC) Date and Time is sufficient proof against intruder. Figure 5 describes working of this module.

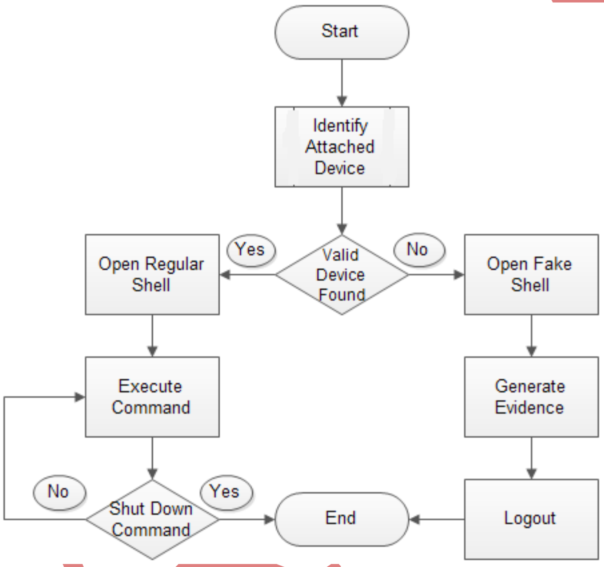


Figure 5: Flow of Trap for Intruder

4.5 Module 5: System Recovery

It is mandatory to set the recovery key immediately after device registration process in order to overcome from the situation, where authenticated user loses the registered device. In this condition, system will treat genuine user as intruder. Module 5 is implemented to solve this issue. Genuine user need to execute recovery module. 'fake_shell' allows to execute recovery module, hence can be used as rescue operation as shown in figure 6.

4.5.1 Recovery Key Generation

For setting recovery key, user will have to answer five recovery questions. Storing the answers of recovery questions is perilous from security concern. Cryptographic hash functions is applied to these answers, and generated hash value is entered into Device Registration File as 'Recovery Key' for specific device. Structure of Registration file is shown in figure 2. Various Algorithms are available to implement cryptographic hash function and their analytical comparison is given in Table 1. Though SHA-256, SHA512 are more secure than MD5, from time and space complexity point of view, MD5 is used in module.

Function	Digest Size (bits)	Rounds	Attacks (Complexity : Rounds)
			Collision
RIPEMD	128	48	2^{18}
RIPEMD 160	160	80	2^{51} 48
SHA 256	256	256	$2^{65.5}$: 31
MD-5	128	64	$2^{20.96}$
GOST	256	256	2^{105}

Table 1: Comparison of various Cryptographic hash functions

4.5.2 Rescue Operation

Once the recovery key is generated, it is easy to get recovered from device lost. User has to execute the recovery module. If answers matches with previous answers then only same hash value will be generated, which is being compared with stored recovery key. If key matches, then registration entry for old device will be removed and services already set for lost device will be deactivated. Intruder can misuse the provided recovery facility, so module is designed to be executed only constrained number of times.

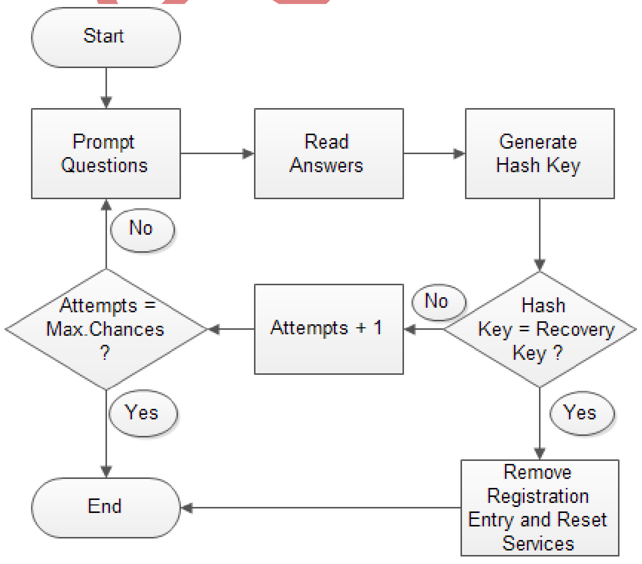


Figure 6: Flow of Recovery Module

V. RESULTS

From the experimental results, it is observed that, as the proposed scheme provides two level authentication, it minimizes the risk level of attacks. Table 2 summarizes various schemes and risk level corresponding to each scheme, where H – High Risk, M – Medium Risk, L –Low Risk and NA – Not Applicable.

Attack / Pass-word Policy	Brute Force	Dictionary	Guessing	Login Spoof	Replay	Shoulder Surfing	Synthetic Template Generator
Weak Alpha-numeric	H	H	H	H	H	H	NA
Strong Alphanumeric	L	L	L	H	H	H	NA
Biometric	L	L	L	L	L	L	H
Graphical	M	L	H	NA	NA	H	NA
Proposed Scheme	L	L	L	L	L	L	NA

Table 2: Risk Levels of possible attacks on various password security schemes.

VI. CONCLUSION

Relying on username-password mechanism for security is not enough. Use of removable storage device instead of biometric authentication system is cost effective. Idea presented in this paper give relief to remember password, change password frequently and secure the password itself. In future, this approach can also be implemented as network and cloud security mechanism.

REFERENCES

Journal Papers:

[1] Mridul Sankar Barik, Gaurav Gupta, Subhro Sinha, Alok Mishra, Chandan Mazumdar. "An efficient technique for enhancing forensic capabilities of Ext2 file system". Elsevier journal- Digital Investigation 4S, Page no. 55-61, Issue June 2007.

[2] Suvrojit Das, Arijit Chattopadhayay, Dipesh Kumar Kalyani, Monojit Saha. "File-System Intrusion Detection by Preserving MAC DTS: A Loadable Kernel Module Based Approach for LINUX Kernel 2.6.x" CSIIRW'09, Oak Ridge, Tennessee, USA. ACM 978-1-60558-518-5, Issue Date: 15 April 2009.

[3] Md. Sarfaraj Alam Ansari, Suvrojit Das, Arijit Chattopadhayay "A Kernel level VFS logger for building efficient file system Intrusion Detection System", IEEE International conference on Computer and Network Technology, Digital Object Identifier : 10.1109/ICCNT.2010.47 Issue Date : 25 April 2010

[4] Hofmeyer, A. Steven, Forrest Stephanie, Somayaji Anil, “Intrusions detection Using Sequences of system calls”, Journal of Computer Security archive, Volume 6 , Issue 3 (August 1998) table of contents Pages:151-180, Year of Publication:1998 ISSN:0926-227X

[5] Bernaschi Massimo, Gabrielliemanuele, Mancini.V Luigi, “REMUS:A security Enhanced Operating System, ACM insactions On Information and System Security, Vol. 5, No. 1, feb 2002, Pages 36-61.

Books:

[6] Bach Maurice J. “The design of the UNIX operating system” Prentice Hall of India; 1988.

[7] Linux Kernel Development (3rd Edition) –Robert Love

Website Links:

[8]http://www.auto.tuwien.ac.at/~chris/research/doc/infsec05_hids.pdf

[9]http://www.reactivated.net/writing_udev_rules.html

[10]http://en.wikipedia.org/wiki/Software_protection_dongle

[11]http://www.hackingalert.net/2011/10/top-10-hacking-softwares-of-2011.html

[12]http://www.instructables.com/id/Bypass-BIOS-Boot-or-OS-Login-to-%22most%22-any-compute/

[13]http://hackersnewsbulletin.com/2013/09/new-password-cracking-software-tries-8-million-times-per-second-crack-password.html