Research and Application of the Transparent Data Encpryption in Intranet Data Leakage Prevention

Conference Paper · January 2009

DOI: 10.1109/CIS.2009.107 · Source: DBLP

4 authors, including:

Ting Chen
University of Electronic Science and Technology of China
62 PUBLICATIONS **1,757** CITATIONS

SEE PROFILE

Research and Application of the Transparent Data Encpryption In Intranet Data Leakage Prevention

Zhang Xiaosong [1], Liu Fei [1], Chen Ting [1], Li Hua [2]
[1] School of Computer Science & Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, China, 610054
[2]Unit 78155 of PLA
Johnsonzxs@uestc.edu.cn, fancylf@sina.com

***Abstract*—Traditional data leakage prevention strategies encompass access control, audit logon and authority management. They have some effects on preventing sensitive information from leakage, while the system's availability may be degraded seriously by their limitation. To solve this problem, a novel model based on Windows file system filter driver is proposed in this paper, in which system encrypts files automatically according to encryption strategies. Leaking confidential information out of enterprise environment, intentionally or unintentionally, will be prevented effectively. Moreover, it has the characteristic of mandatory and transparent encryption, which can compromise the contradiction between data security and user flexibility.**

Keywords-component;File system filter driver; intranet security; transparent encryption; data leakage prevention

I. INTRODUCTION

With the improvement of informationization level, more and more enterprises are conscious of the importance of the intranet security. Usually, a variety of means such as IDS, IPS, firewalls and VPN are utilized to ensure that intranet can work correctly. However, they only defend attacks from external network; because all these measures are based on an assumption that internal network is reliable. However, the reality is often not like this. According to the survey of FBI/CSI, more than 80% attacks come from internal staffs. They may bring viruses into intranet with mobile disk, and take sensitive information away from intranet. Actually, if no effective measure is devised to manage the whole internal network, intranet is unreliable and uncontrollable.

More aspects should be included to construct a trustworthy intranet network, besides the points we discussed above, for instances, detecting unknown and unregistered clients, locating the client who has not updated the latest packets, managing all clients in the intranet network and so on. Among other things, data protecting strategy plays a vital role in intranet security, as intellectual rights are the core of an enterprise.

Generally, most of enterprises will take some measures to prevent sensitive information from leakage, for example, forbidding network connections, forbidding operating mobile storage devices, forbidding using floppy disk, CD-ROM, printers and so on[1]. All these measures can guarantee the security of confidential data, while the availability of the system may be tampered to some extent.

Based on Windows file system filter driver, we proposed a novel model for Data Leakage Prevention (DLP), which works at kernel mode and encrypts data automatically. All confidential data are protected transparently without the consciousness of users. The initial experiments results showed that when applying this model in intranet security protection, it did not degrade the usability of system under the premise of keeping data secure.

II. CURRENT RESEARCH ON DLP

Many technologies are employed for data leakage prevention, including identity authentication, access control, operations tracks and so on.

Wang Lei, Zhuang Yi and Pan Long-ping [2] presented a Mandatory Access Control Model based on Bell-Lapadula model and Bila model. By using Windows file system filter driver and loading MAC strategy, the authors implemented a prototype of monitoring and controlling the whole file system. On the other hand, ZhaoYong *et al*. brought out an intranet information disclosure defendable security model based on crypt-isolation, in which all sensitive information within the intranet environment would be encrypted [1]. Finally this model would form a virtual classified network of password isolation. However, some insensitive information maybe encrypted as well, which impaired the efficiency of the system. To solve this problem, the selective encryption techniques have been introduced in an advanced model by Shufen Liu *et al.* [4]. Only the data sending to different routing domains needed encryption. The key point of this model is to guarantee senders should have the valid identity of the recipient, which maybe impaired by ARP cheating or middle hand attacking.

The researches mentioned above are all trying to resolve the common problem, which is how to balance security, convenience and performance. But at least three aspects of this problem remain unsolved.

1) Temporary file leakage. Some software, e.g. MS Word, will generate temporary files which all data read from and write to actually. The handing of them is often ignored, which is the hidden trouble.

2) Partial disk encryption. Users should not be limited to encrypt data transparently in special directories, but

978-0-7695-3931-7/09 $26.00 © 2009 IEEE
DOI 10.1109/CIS.2009.107

global disk confidential data are protected no matter which disk or directory they appear in.

3) Encryption based on postfix. Files are often encrypted if they satisfy specific conditions, such as the postfix or the format, which may consist of the leakage origin when trusted processes save files as undefined format file.

III. PROTECTION DATA WITH TRANSPARENT ENCRYPTION

Windows file system is designed to be an open architecture. It is allowed to add all kinds of drivers, which enriches the functionalities of file systems. And file system filter driver is the typical one.

File systems such as NTFS, CDFS, and FASTFAT correlate a control device object, which implement the functions of configuring file system drivers as well as connecting with applications. And volume device objects such as "HardiskVolume1" are mounted to a specific file system. All IRPs (IRP, I/O Request Packet) are actually sent to these volume devices. Therefore, through attaching devices to volume devices, the IRPs will be intercepted before they reach their intended target [5][7], and then the filter driver will have opportunities to modify the functionality provided by the original IRP.

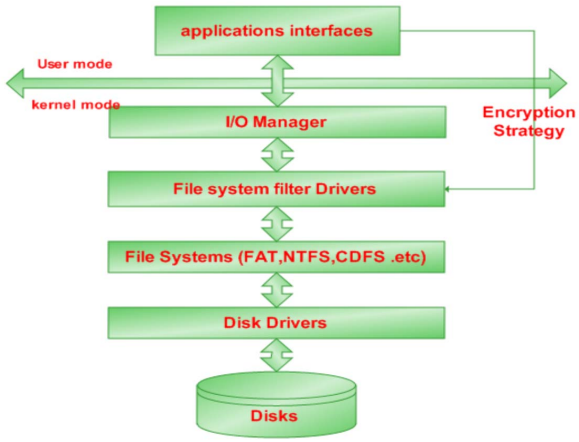

Figure 1. File System Filter Driver

Based on the discussion above, transparent file encryption is developed. When writing data into storage devices, the IRP_MJ_WRITE IRP is sent to a volume device, which will be caught first by filter driver. According to the encryption strategy, the filter driver can encrypt data contained in this IRP, changing data from plaintext to ciphertext, and then pass the modified IRP to the volume device. On the other hand, when reading data from storage device, the IRP_MJ_READ IRP will be received by volume disk, and encrypted data read from storage devices will be decrypted by filter driver, which converts ciphertext into plaintext and then sends plaintext to applications in user mode. This is how transparent encryption works.

When transparent file encryption technique is applied in data leakage prevention, it will be convenient not only for user operations, but also protecting data from leaking out effectively with its two remarkable characteristics:

1) Transparency: contrary to traditional encryption, it runs in kernel mode. Systems will encrypt and decrypt files automatically without users' intervene. It seems that there is no difference for users between operating encrypted files and unencrypted ones. All operations are transparent for users, which indeed bring users many conveniences.

2) Coerciveness: the action of encrypting is forcefully done by system regardless of users' agreement. All files satisfying the encryption strategy are saved as cryptograph no matter which medium they are stored in. And all data sent to trusted applications will take on in the form of plaintext (fig.2 shows the procedure). Many facts indicate that only mandatory encryptions can prevent confidential data from leaking.

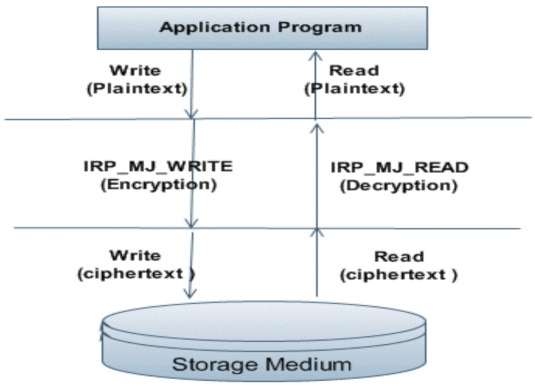

Figure 2. Transparent Data Encryption

IV. THE REALIZATION OF THIS MODEL

Because the kernel APIs (API, Application Programming Interface) depend on OS largely in file system filter drivers, the judgment of OS is indispensable before realizing the whole project [6].

A. *The composition of this model*

The whole model is composed of four parts, which are Application Control module, Identity Authentication module, Security Strategy Management module and Windows file filter driver module. Each module realizes the functions as follows:

1) Application Control module. This module is responsible for accepting authentication, sending control code to filter drivers, constituting security strategies, receiving the result of authentication and kernel operation.

2) Identity Authentication module. Before a user logons to system, this module realizes the algorithm of identity authentication.Identifying whether he is a legal one or not is crucial to the whole system.

3) Security Strategy Management module. This module receives and stores the strategies from users, and post security strategies to filter driver to implement.

4) Windows file filter driver module. This module is the core of the whole system. It implements all kinds of security strategies set by users and realizes the functions which are access control, transparent data encryption, real-time monitoring.

These three functions are actually working at different stages which are before, at and after the incident. Access control and real-time monitoring are passive protection methods, while transparent data encryption is independent on user's will, which prevents data from leakage intentionally or unintentionally. The implement of it is a hard work .And we will discuss the details in this paper.

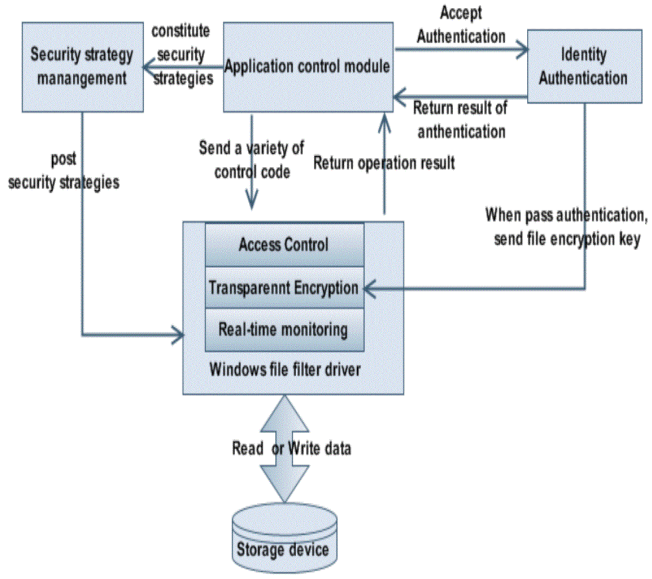

Figure 3. Data Protection Model

B. Three types of device object

There are totally three types of device object generated by file system filter drivers. The table below shows the detail information of them.

TABLE I. THREE TYPES OF DEVICE OBJECT

Type	*Generation time*	*Functions*
Control device object (CDO)	In DriverEntry routine when loading drivers	Communicate with applications and configure filter driver
Filter device object attached to CDO of file system	When a new file system has been loaded to system	Catch IRP of IRP_MJ_FILE_SYSTEM_CONTROL to dynamically attach to new volume device
Filter device object attached to volume object	When a new volume has been mounted to a specific file system	Filter all IRPs of file operation and realize the core functions of transparent encryptions

C. Realization of encryption and decryption

After volume devices have been attached by our devices, all file operation IRPs will be filtered, and the driver mainly handles two dispatches which are IRP_MJ_WRITE and IRP_MJ_READ. The typical procedure of encryption is as follows.

1) Check the way of reading and writing, and make sure that the flag of IRP is one of IRP_NOCACHE,IRP_PAGING_IO,IRP_SYNCHRONOUS_PAGING_IO. Get the data buffer Address in original IRP, and extract data as the plaintext.
2) Allocate a buffer address from NonPagedPool to keep plaintext.
3) Call IoAllocateMdl and map the buffer allocated just now to renew original IRP's MdlAddress.
4) Choose a cryptography algorithm such as RC4, AES, DES to change plaintext to ciphertext.
5) Set completion routine, and call IoCallDriver to pass the modified IRP to lower devices.
6) In completion routine, set IRP using the original Mdldress and UserBuffer.

The usual procedure of decryption is largely similar to encryption except that the data should be decrypted in completion routine for the reason that ciphertexts should be gotten first from disk storages.

D. Flushing the cache

In the Windows NT system, the way of reading and writing includes Cached I/O、Non-cached user I/O、Paging I/O. And it is not advisable to encrypt the cached data. Therefore, data coming from the way of Cached I/O are not allowed to handle. In other words, we should encrypt data using the way of Non-cached user I/O and Paging I/O. Just for this reason, it is ensured that all data in disk will be saved as cryptograph while all data in cache memory keep the state of plaintext. When ciphertext stored in disk are read by an application for the first time, the Cache Manager will cache plaintext in memory. At this time when another application is to read the same data, it will read data directly from cache memory and then plaintext will be copied away, which obviously bring security problems. To resolve this issue, one method can be employed which is flushing cache memory when the file is closed. And there exists a series of kernel APIs with the prefix of "Cc" for cache memory operations. For example, the "CcFlushCache" routine flushes all or a portion of a cached file to disk [7].

E. Intermediate file encryption

Some program will generate temporary files during editing and all I/O operations are actually based on these files [3]. When action of closing happens, original file will be deleted and intermediate file will be renamed to have the same name as original file. If files are encrypted in terms of postfix or file format, intermediate files will not be encrypted probably, which leads to final files being saved as plaintext in disks. Therefore, encrypting files without identifying the postfix of files should be put into use. In order to gain this goal, there are at least three steps can be followed.

1) Trusted processes should be included in encryption strategies.
2) Catch all IRP_MJ_WRITE IRPs. If they derive from trusted processes, go to the next step. Otherwise let them pass through.
3) Encrypt data contained in IRP_MJ_WRITE IRPs.

F. *Identifying encrypted files*

This is a troublesome issue, for the reason that no reasonable solution exists until now. However, identifying which file has been encrypted and which file needs to be decrypted is vital for developing the transparent encryptions drivers. Adding encryption tag to original file in head or tail is a usual method. These two ways are comparatively more reliable when considering the whole system's stability and compatibility.

For example, in the routine of IRP_MJ_CREATE, filter driver extracts the file object from IRP stack, and identifies whether this file has encryption tag. If it does, delete it and add its reference counting which is contained in global file context chain. And in the routine of IRP_MJ_CLEANUP, Get the file object from IRP stack again, and search file reference counting from file context chain. Decrease it. If reference counting has reached zero, add the encryption tag in tail to the file which satisfies the encryption strategy.

Usually, the encryption tag is a data structure, which consists of several parts. They are feature string, encryption algorithm index, generator name, company name and so on.

G. *Encryption key management*

According to the characteristic of symmetric encryption and asymmetric encryption, we symmetrically encrypt the file data and asymmetrically encrypt the symmetric encryption key. In this way, it guarantees the security and efficiency of data processing [8]. The generation and storage of keys in symmetric encryption and asymmetric encryption are key points of the whole system.

When users register to the system for the first time, the conjunction of user name and password can be hashed to 128 or 256 bits to act as private key for symmetric encryption(PKSE), and randomly generated a pair of key which are private key(PRKAE) and public key(PUKAE) for asymmetric encryption. PKSE will be encrypted with PRKAE. And the result together with PRKAE is to be stored in a mobile media to form USB key while PUKAE can be configured to host system. Before user attempt to logon system, inserting USB key and identity authentication are needed. Only if the correct user name and password have been input, can the system get PKSE with PRKAE which is taken from the USB key. In fact, before user logon on system, two factors are required. One is the pair of user name and password, and the other is the USB key. In this way illegal logons are largely avoided.

V. CONCLUSIONS

In this paper, we presented a model of data leakage prevention based on Windows file system filter driver, and analyzed why it can balance the data security and user convenience. Furthermore, we've shown the implementation details of our prototype system. Undoubtedly, Windows file filter driver, especially technique of transparent data encryption which provides a technical advantage to system compatibility and operation facility, will be applied more and more in intranet security. In addition, it can be used in other fields for data protection, for instance, database [9]. However, it is far less than needed for constructing a reliable intranet network which asks for the holistic and integrated strategies. Our future work will focus on client monitoring and control, network management and operation audit.

ACKNOWLEDGMENT

This work is sponsored by Science and Technology Commission of Shanghai Municipality (NO. 09511501600).

REFERENCES

[1] Zhao Yong, Liu Jiqian, Han Zhen,Shen Changxiang, "The Application of Information Leakage Defendable Model in Enterprise Intranet", In: Journal of Computer Research and Development, pp 761-767 2007 44(5)

[2] Wang Lei, ZHUANG Yi, Pan Long-ping, "Design and implementation of file watching system based on mandatory access control", In: Computer Application. Vol.26 No.12 Dec.2006

[3] Lei Zheng, Zhao-feng Ma, Ming Gu, "Techniques of File System Filter Driver-based and Security-enhanced Encryption System", In: Journal of Chinese Computer Systems.Vol28.no.7,July 2007

[4] Shufen Liu, Zhagxiang Zhang, Yaorui Cui, Lin tao Wu;"A New information Leakage Defendable Model " In: Computer-Aided Industrial Design and Conceptual Design, 2008. CAID/CD 2008. 9th International Conference on,pp:109-112, Nov. 2008

[5] Rajeev Nagar, Windows NT file system internals: a developer's Guide. O' Reilly, 1997

[6] DAI Xing-ke , SHE Kun , SHANG Qing-hong, "Data Secure Storage Based on File System Filter Driver", In: China information Security ,Vol.8,2007

[7] Microsoft. Corporation; "Installable File System Kit Document[S]. Microsoft Corporation", 2002.

[8] Microsoft Corporations: "Using Encrypting File System", published: November ,2005

[9] Ulf T. Mattsson, CTO Protegrity, "A Practical Implementation of Transparent Encryption and Separation of Duties in Enterprise Databases, Protection against External and Internal Attacks on Databases", In: E-Commerce Technology, 2005. CEC 2005. Seventh IEEE International Conference on, pp:559–565,19-22 July 2005