ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 6, No. 3, 2011, *pp*. 163-172

Full Disk Encryption based on Virtual Machine and Key Recovery Scheme

Min Liang [1, +], Chaowen Chang[1]

[1] Institute of Electronic Technology, Information Engineering University, Zhengzhou China

(*Received November 18, 2010, accepted December 20, 2010*)

Abstract. Full disk encryption is a good choice to solve the problem of information leakage. In this paper, a full disk encryption based on virtual machine is proposed for computers without TPM. The program to decrypt the operating system is stored in a USB device which is more secure than in hard disk together with confidential information. Its key recovery scheme is with the help of a smartcard which is used for enhancing security. The experiments, security analysis and comparison demonstrate the efficiency, security and advantage of the proposed scheme.

Keywords: Full Disk Encryption, Virtual Machine, Key Recovery, Smartcard

1. Introduction

Storage security is one of the important preconditions for information security and information leakage from the disk has been a serious problem. Encryption is often used in protecting information from modification and leakage, but it is not secure only encrypting the information itself. Full Disk Encryption (FDE) is an effective manner to protect the confidential information in the hard disk. It is widely used in governmental and military departments. FDE can encrypt all the data including the operating system files, temporary files and user's files. And Microsoft has provided Bitlocker Drive Encryption Technology but there are some limitations. FDE provides a comprehensive protection for the confidential information. However, there must be a program to decrypt the operating system files, if not, the system could not start. But it is not secure that the program and confidential data are stored together in hard disk. USB device is a good choice to store the program.

The security of cryptographic data is based on the key which is used to encrypt and decrypt data. The key should be stored in a safe place, but also should be convenient to use. The hardware token, such as smartcard, is a good solution. But there must be some methods to recover the key when the smartcard is lost or stolen, and maybe the key is damaged.

The secret is only accessed by limited users, who are based on the authorization, but not the position in organizations or something else. We call this private hierarchy. The key should be recovered by the authorized user, not by the system administrator or the administrative superior.

In this paper, we present full disk encryption based on virtual machine (VM) and its key recovery scheme based on Shamir's secret sharing scheme. In this scheme, we use smartcard to store user certificate and confidential information such as the privacy key. The program, XEN, is stored in a USB device instead of in the same place with confidential data. The key is stored nowhere and generated in the startup. The key can be recovered without the administrator knowing it, when the user lost his smartcard and USB device.

The roadmap of this paper is as follows. We present the related work in Section 2. A detailed review of Bitlocker is provided in Section 3. We present the full disk encryption based on XEN in Section 4 and the key recovery processes in Section 5. We make an evaluation in Section 6. We draw a conclusion in Section 7.

2. Related Work

Full disk encryption uses disk encryption software or hardware to encrypt every bit of data that goes on a

[+] Corresponding author. Tel.: +86-0371-8163 8719.
E-mail address: lm7186345@163.com.

disk or disk volume. Full disk encryption prevents unauthorized access to data storage. There are multiple tools available in the market that allow for full disk encryption. They are divided into two main categories: hardware-based and software-based. The hardware-based full disk encryption solutions are considerably faster than the software-based solutions, but more expensive. Bitlocker is a software-based solution which is available in some editions of Windows.

Many kinds of key recovery scheme have been proposed in [1], [2], [3], [4], [5] and [6]. The main key recovery methods are key escrow, trusted third party, key backup and key encapsulation. Starting in 1993, the US government announced a new encryption technology call key escrow system [7]. Key escrow is an "all or nothing" proposition, with no mechanism to guarantee that the caretaker is doing the job honestly [8]. Most key recovery systems based on trusted third party aim to recover lost keys, support law enforcement for message investigation, and consider personal rights of privacy [9]. Key backup is a simple method which is used in Bitlocker. Key encapsulation is the only one in which the key is not known to the administrator [4, 6]. It is hard to confirm that the recovered key is the legitimate user's key. This means that the key can be recovered by a malicious user. The key with a certificate can resolve the problem, but the key would be known to the administrator in advance. Key recovery schemes using a smart card have been proposed in [10] and [11], but they are different from the one in this paper. Shamir secret sharing scheme is one of the key recovery solutions proposed by Shamir [12], also referred as (t, n)-threshold secret sharing scheme. In order to reconstruct the secret, one only obtains any t of the n shares. There is a scheme that integrates the (t, n)-threshold scheme into document archiving system to recover key [1].

There is a key recovery scheme which is used in FDE [11]. In that method the author uses a smartcard to store cryptographic object and makes use of a blind signature for both the generation of the disk encryption key and authentication. It can limit the recovered encryption key without informing the administrator. But the decryption program is stored together with the confidential data in hard disk, which is not secure.

3. Bitlocker

Bitlocker is a typical full volume encryption scheme. Bitlocker is available only in the Windows Server 2008, in the Enterprise and Ultimate editions of Windows Vista and in Windows 7. There are at least two NTFS-formatted volumes: one for the operating system and another called the system volume with a 1.5GB from which the operating system boots. Bitlocker requires the system volume to remain unencrypted. Bitlocker encrypts the entire Windows operating system volume on the hard disk [13]. Volumes other than the operating system volume and the system volume are called data volumes. Bitlocker encryption of data volumes is supported only in Windows Server 2008 [14]. So Bitlocker is not a strict sense of full disk encryption.

Bitlocker provides the most protection when used with a Trusted Platform Module (TPM) version 1.2 which is installed in many newer computers by the computer manufacturers. The key used for disk encryption is sealed by TPM and will only be released to the OS loader code if the early boot files appear to be unmodified. But this mode is vulnerable to a cold boot attack, as it allows a powered-down machine to be booted by an attacker. In addition to the TPM, Bitlocker provides user authentication mode. This mode requires that the user provide some authentication to the pre-boot environment in the form of a pre-boot personal identification number (PIN) or inserting a USB flash drive that contains a startup key. This mode is vulnerable to a bootkit attack. However, there are still a large number of computer without TPM. On computers without TPM, Bitlocker encrypts the Windows operating system drive requiring the user to insert a USB startup key to start the computer or resume form hibernation, and it does not provide the pre-boot integrity verification offered by TPM. This mode is also vulnerable to a bootkit attack.

Furthermore, the key recovery of Bitlocker is simple and not secure. In Bitlocker, recovery consists of decrypting the volume master key using either a recovery key stored in the form of plaintext on a USB flash drive or a cryptographic key derived from a recovery password. The TPM is not involved in any recovery scenarios. The recovery password can be printed or saved to a file and the recovery key can be created and saved to a USB flash drive during Bitlocker setup. A domain administrator can generate recovery passwords automatically and transparently back them up to servers. There is not enough protection to the password and key which are plaintexts. The adversary can get access to the encrypted data as long as they get the password and key.

In short, the shortcomings of Bitlocker are as follows:

1) The scope of Bitlocker is limited. It is available in Windows Vista and Windows 7, but not supported

in Windows XP which is widely used, Linux and Mac OS, etc.

2) Bitlocker is not a strict sense of full disk encryption. The system volume is not encrypted and Bitlocker encryption of data volumes is not supported in the Enterprise and Ultimate editions of Windows Vista and in Windows 7.

3) The TPM only mode is vulnerable to cold boot attack. User authentication mode and USB mode without TPM are vulnerable to bootkit attack.

4) The key recovery is simple and less secure. The administrator can gain the confidential information encrypted by Bitlocker by key recovery ways.

4. Full Disk Encryption based on Virtual Machine

FDE is often used in the organization of high security requirement. Because all of data including the operating system files in the hard disk is cryptographic, there must be a program to decrypt it. Usually, the program is stored in plaintext with the cryptographic data together in the disk. The program may be tampered with. For example, a malicious user could use a Windows PE CD to access and falsify the program code in the form of plaintext. So, the confidential data may be leaked. In our scheme, the program is XEN [15] stored in a USB device, and we call it XEN-USB. XEN-USB is separated with the encrypted hard disk. We can not take PC to everywhere, but we can do so with USB device. The XEN is more secure in the USB device than in the hard disk of PC which is not easy to protect. The operating system in the form of ciphertext is running after XEN-USB decryption process.

The smartcard is capable of storing cryptographic objects, such as the private key of a key-pair, digital certificates, and symmetric encryption keys. In addition, it is also capable of carrying out cryptographic functions, such as symmetric and asymmetric encryption and calculating hash functions, etc. In this scheme, the reason for employing smartcard is that smartcard is both secure and convenient for storing the user's private key and certificate. Furthermore, smartcard and XEN-USB can take a two-factor authentication, which is more secure.

Our scheme is called a VM-based full disk encryption (VMFDE). Encrypted OS is running after inserting XEN-USB to complete pre-boot process. Its overall structure is shown in Figure1. The goal is to ensure that all stored on the hard drive data is the form of ciphertext. The attacker in the absence of security Flash disk case, has no way to access the hard drive and steal the data, and can not get any information from the encrypted hard drive.

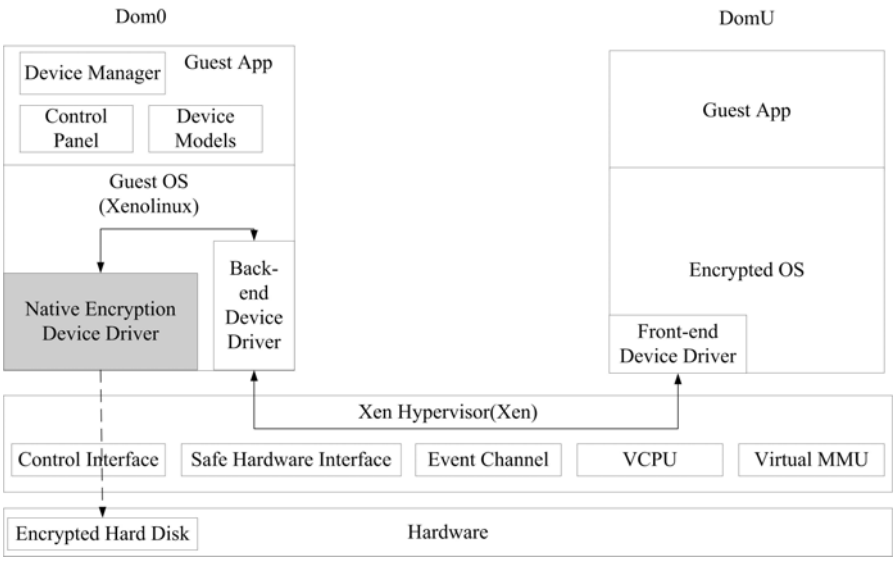

Fig.1: Full Disk Encryption based on XEN

XEN is a virtual machine monitor (VMM), which directly runs on the naked hardware located in the ring0. XEN can control, manage and virtual hardware resource, schedule the virtual machines and control the access to shared resources. Dom0 is a privilege domain whose kernel runs in the ring1. XEN and Dom0 are located in XEN-USB and the encrypted OS is in the hard disk of PC. XEN and Dom0, in the XEN-USB,

start after the system self-test and the BIOS loading. Later, the entire system including the CPU, memory and hard disk, is under the management and control of XEN. Through the control panel, entering “xm” starts an encryption OS on the hard disk. Hard disk encryption and decryption procedures are in the native encryption device driver, and the decryption process automatically decrypts OS on the hard disk to achieve a transparent encryption and decryption process.

The encryption and decryption processes are completed in the native encryption device driver in the Dom0. Front-end driver in the Encrypted OS transfers the I/O request to the back-end driver in the Dom0, and the back-end driver parses the incoming I/O request and maps to the native encryption device driver. Finally, the encryption and decryption are completed by the native encryption device driver. Of course, all of these are carried out under the control of the XEN. The system resources are controllable by XEN. Encrypted OS accesses the ciphertext in the hard disk through XEN and Dom0. The paging files, temporary files and all system files arising from encrypted OS are stored in the form of ciphertext in the hard disk.

5. Key Recovery Scheme

5.1. Overview

In this section, we describe the key recovery environment and scenario. We assume that the scheme is used in an organization such as military departments.

The key, which is mentioned in this scheme, is the *main key* (MK), not the encryption key. If we only use one key to protect all the data in the disk, it is definitely not secure [16]. So the key should be arranged in layers. MK is used to protect file folder keys which is generated by the attribute of file folder. The file in the disk is encrypted by AES with the encryption key which is protected and generated by file folder keys.

The *key management server* (KMS) is used to generate the certificate and recover MK. It is also capable of storing confidential information such as its privacy key and the shares.

The assumption and precondition of our scheme are as follows:

1) The scheme is used in a secure and controllable network such as an organization or military network.

2) The transference between smartcard and XEN-USB (or KMS) is not monitored and tampered with.

3) The XEN-USB and smartcard are secure and tamper resistant. The cryptographic object can be not stolen from both.

4) The KMS is reliable and can prevent the confidential information from leaking out.

5) The trustees are dependable and independent. They will not decrypt shares without authorization.

6) The scheme is aimed at the situation that the user has lost the smartcard and XEN-USB, but does not forget its password.

To recover MK, we design our scheme which is based on (t, n)-threshold and RSA. We use a smartcard to store cryptographic objects as a result of improving security. Also XEN, the decryption program, is stored in USB device in order to prevent being falsified. There are four main components in the system: 1) KMS, 2) the smartcard, 3) XEN-USB, 4) the FDE PC.

KMS manages shares for reconstructing the secrets which is related to MK recovery. The shares are not stored in plaintext form. They are encrypted by the public keys of trustees. So, shares must be decrypted by at least t trustees using private keys during MK recovery processes.

5.2. Key Management Server

Based on the design described above, KMS is composed of 4 modules: (1) user initialization module, (2) share management module, (3) digital certificate management module, and (4) key recovery module. Fig. 2 gives an overview of KMS.

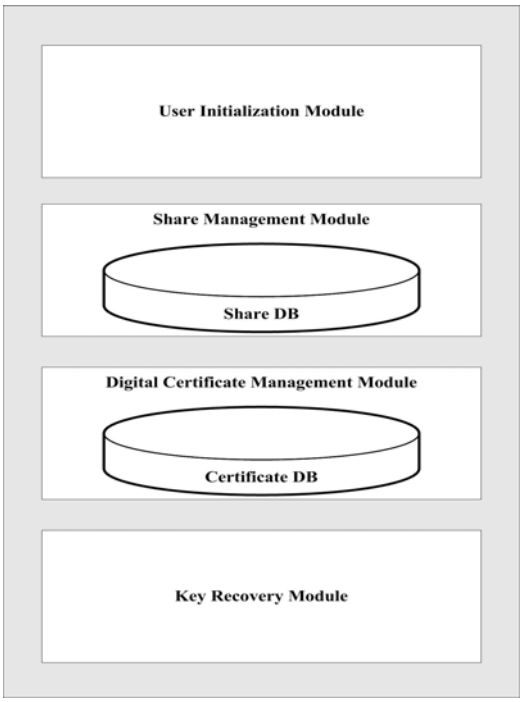

Fig.2: An Overview of KMS

The user initialization module is responsible for the initialization process for new users. It completes the smartcard registration and XEN-USB registration processes, generates the certificates and secrets, and deposits these shares into the share management module. Shares are not in plaintext form. Rather, they are encrypted using the public key of trustees during the registration process.

The share management module manages shares for reconstructing the secrets which is used to recovery key. It is a database storing all these shares. When a key needs to be recovered, the corresponding trustees can receive and decrypt the shares with their private keys.

The digital certificate management module is a database that keeps a copy of certificates of all members in the organization.

The key recovery module is used to manage and complete the recovery process. It sends the encrypted shares to trustees and receives the decrypted ones. The secret is reconstructed in this module.

5.3. Processes

The definition of notations is as follows:

U_i : *User i*
pw_i : U_i'*s password*
(e, d, n) : *RSA keys of KMS*
(e_i, d_i, n_i) : *RSA keys of* U_i
R, r_i : *Random numbers* $(\leq |n|)$
a_i, b_i : *Random numbers* $(\leq |n|/2)$
$cert_i$: *Certificate of* U_i
$cert_N$: *Certificates of N trustees*
$CERT$: *Certificate of KMS*
c : *Challenge*
mk_i : *MK of* U_i
$h(.)$: *Hash function*
N : *Number of trustees*

1) Initialization Process

KMS generates its public and private keys, CERT and R. It appoints which entities are trustees. It also

generates and stores their certificates.

2) Smartcard Registration Process

U_i connects his smartcard to KMS. KMS generates cert_i. U_i sets and store pw_i in his smartcard. $R^{d_i} (\text{mod}\, n_i)$ is U_i's public parameter (Fig. 3).

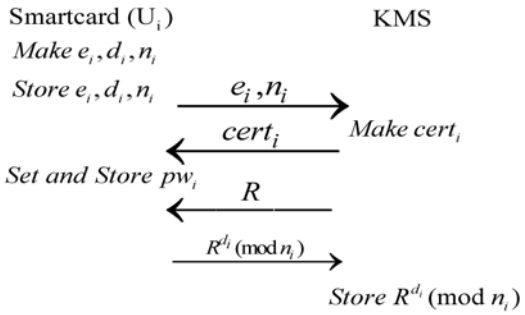

Fig.3: Smartcard Registration Process

3) XEN-USB Registration Process

XEN-USB is connected to KMS with a Secure Sockets Layer. $R^{d_i} (\text{mod}\, n_i)$ is generated by KMS and stored in the XEN-USB. S denotes $(a_i b_i)^e (\text{mod}\, n)$ which is split n shares. XEN-USB encrypts n shares using the public keys of n trustees and transmits them to KMS (Fig. 4).

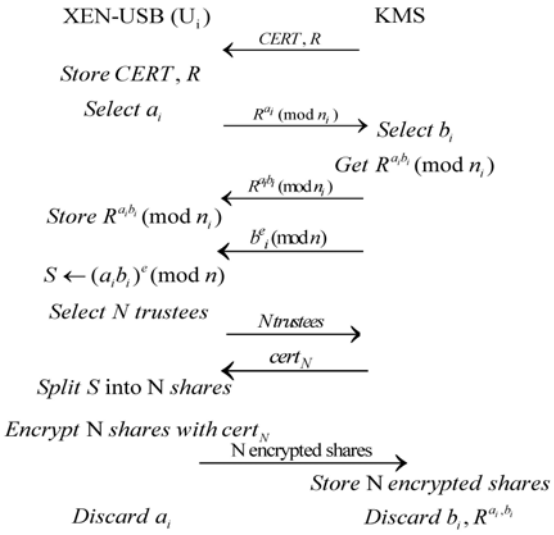

Fig.4: XEN-USB Registration Process

4) Authentication Process

XEN-USB confirms the truth of smartcard by the way of communicating with KMS via the network and verifying the certificate. The user's validity is approved by the password. With the help of c, it completes the authentication using a blind signature (Fig. 5).

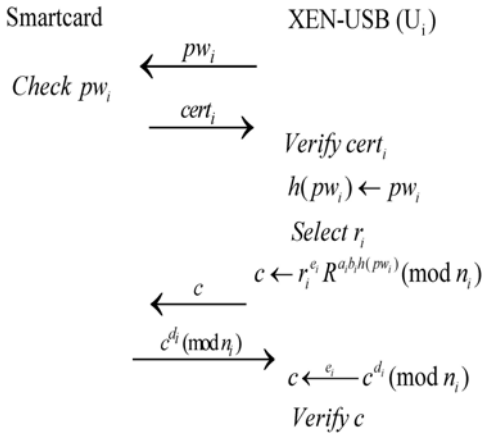

Fig.5: Authentication Process

5) MK Generation Process

The MK generation is done after the authentication process. XEN-USB calculates the follow formula after verifying c.

$$mk_i = c^{d_i} / r_i (\mathrm{mod}\, n_i) = R^{a_i b_i (pw_i) d_i} (\mathrm{mod}\, n_i), \quad (1)$$

6) Key Recovery Process

When U_i lost his smartcard and XEN-USB, U_i needs to apply to administrator for a new XEN-USB. U_i connects the new XEN-USB to the FDE PC and sends recovery application to KMS via the network. After receiving the application, KMS notifies the corresponding trustees. As mentioned before, at least t trustees must participate and decrypt the encrypted shares. Once more than or just t have decrypted shares, S can be reconstructed and mk_i can be recovered (Fig. 6).

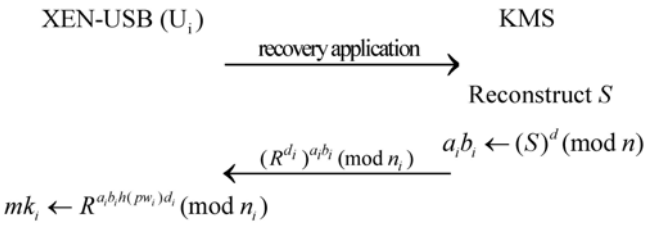

Fig.6 Key Recovery Process

6. Evaluation

6.1. Performance

The performance of this scheme relies on the asymmetric encryption cost, on the smartcard resources and capacity and on the USB device transmission speed. In this section, we measure the cryptographic workload in order to evaluate the performance of our scheme.

PC was a Lenovo Qitian M7000 (CPU: Intel Core2 Quad Q9400 2.66GHz, Memory: 2G, Hard disk: 320G), which CPU supports Intel-XT. XEN can create and run Windows with the help of Intel-VT [17]. We used as the USB device a Kingston Data Traveler 8G, used as the smartcard a CryptoMate (ACS). The virtual machine we used was XEN 3.0 and the encrypted operating system was Windows XP.

Our first experiment was mainly measuring smartcard and XEN-USB cryptographic cost in four processes listed in Table 1. The results show that the XEN-USB load is less than the smartcard for simple reasons depending on RSA itself and calculation capability of smartcard. In fact, the RSA public key operations are much faster than private key operations. The CPU of smartcard is obviously slower than the PC's. The total time before the encrypted OS booting up is less than 2 seconds when the user first use including registration phase, authentication phase and MK generation. After the first time, the time until the

OS has booted up is required for the authentication phase and MK generation phase, and the OS booting. The time, Windows XP booting up without encryption on the XEN, is about 60 seconds on the same PC. So the processing time for first use was less than 5% of the time required for booting up the OS. On the other hand, the time for daily use was less than 1% of the OS booting time. Therefore we think that the cryptographic time is short enough to be acceptable.

Table 1: Cryptographic Operations and Workload

	Smartcard	XEN-USB
Smartcard Registration	Make e_i, d_i, n_i (1750ms)	—
XEN-USB Registration	—	Split S into N shares(40ms),Encrypt N shares(120ms)
Authentication	Generation of blind signature(380ms)	Generation of challenge(17ms),Verification of blind signature(12ms)
MK Generation	—	$mk_i = c^{d_i} / r_i (\mathrm{mod}\, n_i)$ (0.2ms)

Furthermore, for performance evaluation we used IOzone, a free file system benchmark tool, to compare with Bitlocker. IOzone takes various file operations as basic workloads to test the I/O performance of file systems and allows to adopt diverse testing modes from automated to partial. The file operations include read, write, re-read, re-write, random read/write and so on. We have separately run IOzone on Windows XP, Windows Vista with Bitlocker and VMFDE where the encrypted OS is Windows XP. The read and write file operations are evaluated by IOzone in the three systems. The test data size is from 32MB to 1024MB. The result is shown in Fig. 7 and Fig. 8 and the time is in seconds. The experimental data indicates that the performance of VMFDE is close to, but does not outperform Bitlocker. This is partly because XEN is stored in USB device for security concern. Virtualization cost is also another reason. This can be mitigated and even eliminated by hardware development and improvement.

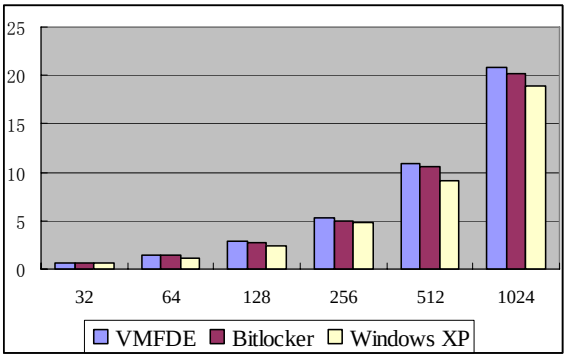

Fig.7: Read Operations of Different Environments and Cases in IOzone

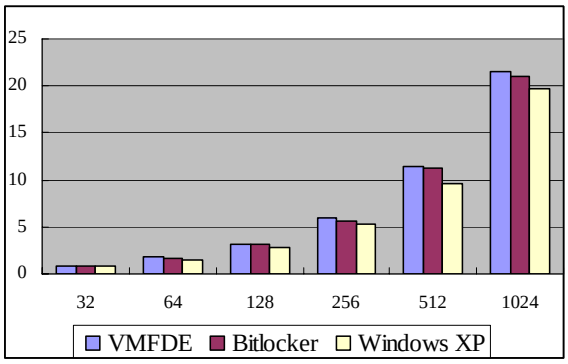

Fig.8 Write Operations of Different Environments and Cases in IOzone

6.2. Security Analysis

In this section, we analyze the security of our scheme. Firstly, XEN is stored in USB device, which is easier to protect and more secure. XEN-USB can be prevented from falsifying. Secondly, KMS stores cert_i, $R^{d_i} (\mathrm{mod}\, n_i)$ and the n encrypted shares. The malicious user can not calculate d_i from $R^{d_i} (\mathrm{mod}\, n_i)$ and $R^{d_i} (\mathrm{mod}\, n_i)$ only can be got by the user who has the smartcard. MK is calculated by three parameters: S, pw_i and d_i. S is split into n shares which are encrypted by the public of trustees. pw_i is known only by the legitimate user. d_i is stored in the smartcard which is tamper-resistant. This means that the scheme can protect MK and limit MK recovery. Lastly, our scheme takes the authentication between XEN-USB and smartcard via a blind signature and verifying user's password.

Cold boot attack involves rebooting a computer which has been handling sensitive information, and dumping contents of its memory out to a disk in order to reconstruct keys. The attack relies on the data

remanence property of DRAM to retrieve memory contents which remain readable in the seconds to minutes after power has been removed. The attack has been demonstrated to be effective against full disk encryption schemes of various vendors and operating systems, even where TPM is used. The problem is fundamentally a hardware but not a software issue. Therefore, ensuring that the computer is shut down whenever it is in a position where it may be stolen can mitigate this risk. In order to protect against cold boot attack against hibernation, VMFDE encrypt the hibernation files and dismount hard disk when hibernation. The sleep mode is generally unsafe. So configuring an operating system to shut down or hibernate when unused, instead of using sleep mode, can help mitigate this risk.

Offline software-based attack can be prevented by VMFDE. Any alternative software that might start the system does not have access to the encrypted data including system files. The attacker without the decryption keys for the operating system drive can not get anything from the disk.

Bootkit is a kind of rootkit which can achieve its function by tampering with the system boot files. Bootkit can replace the legitimate boot loader with one controlled by an attacker without being detected. In VMFDE, bootkit can be detected by XEN. Encrypted OS is monitored by XEN which is running in the ring0. Secure mechanism, such as integrity verification, can be added in XEN. XEN is separated from encrypted OS. As long as the XEN-USB is secure, protected and kept well, bootkit attack can be prevented in VMFDE.

6.3. Comparison with Bitlocker and VMFDE

VMFDE is compared with Bitlocker in the aspects of supported operating systems, pre-boot authentication, layering and the security in Table 2.

1) Operating systems. VMFDE is available in more operating systems than Bitlocker because both Windows and Linux are supported by XEN.

2) Pre-boot authentication. Authentication can be required before booting the computer in Bitlocker and VMFDE.

3) Layering. Bitlocker encrypts the disk except for the boot volume and VMFDE can encrypt the whole disk. Swap space and hibernation file can be protected and encrypted by Bitlocker and VMFDE.

Table 2: Bitlocker and VMFDE

		Bitlocker	VMFDE
Operating Systems		Windows Vista, Windows Server 2008, Windows 7	Windows, Linux and Mac OS, etc
Pre-boot Authentication		Yes	Yes
Layering	Whole Disk	No(except for the boot volume)	Yes
	Swap Space	Yes	Yes
	Hibernation File	Yes	Yes
Security of Key Recovery		Poor	Good
Cold Boot Attack		No	No
Bootkit Attack		Yes(with TPM)	Yes

4) The key recovery scheme of Bitlocker is a key backup method and the key is simply stored in USB devices or printed in the form of plaintext. The key itself of VMFDE is not stored in anywhere and can be reconstructed when recovered.

5) Both Bitlocker and VMFDE can not prevent cold boot attacks. Cold boot attack can be mitigated through secure user modes. For example, the computer is shut down or hibernation in stead of sleep mode. Bootkit attack can be mitigated in VMFDE by integrating security mechanisms into XEN. Bitlocker can mitigate the Bootkit only when the computer with TPM. For computers without TPM, VMFDE is more secure than Bitlocker. VMFDE is much more scalable and flexible than Bitlocker.

7. Conclusion

In this paper, we introduce a full disk encryption based on virtual machine and its key recovery scheme. XEN, which is used to decrypt the confidential data in the disk, is stored in a USB device. This is more secure than stored together with encrypted data in hard disk. VMFDE can be used to protect many operating systems. In this way, we can guarantee the integration of XEN and prevent confidential information from

leaking out. Based on RSA and (*t, n*)-threshold secret sharing, our scheme can recover the key without administrator knowing any confidential information. Furthermore, the experiments and analysis show that the proposed scheme is efficient and secure.

8. Acknowledgements

This work is supported by the National High Technology Research and Development Program ("863" Program) of China (No. 2007AA01Z479).

9. References

[1] Eric K. Wang, Joe C.K. Yau, Lucas C.K. Hui, Zoe L. Jiang, S.M. Yiu. A Key-Recovery System for Long-term Encrypted Documents. *In Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference Workshops*. Hong Kong. 2006, pp. 52-55.

[2] Kanokwan Kanyamee, Chanboon Sathitwiriyawong. A Secure Multiple-Agent Cryptographic Key Recovery System. *In Proceedings of the 10th IEEE International Conference on Information Reuse & Integration*. 2009, pp. 91-96.

[3] Yahya Y. Al-Salqan. Cryptographic Key Recovery. *In Proceedings of the 6th IEEE Workshop on Future Trends of Distributed Computing Systems*. 1997, pp. 34-37.

[4] M. J. Markowitz and R. S. Schlafly. Key Recovery in SecretAgent [Online]. Available: http://www.infoseccorp.com/news/press/pdf/sakr.pdf. 1997, June 11.

[5] Y. Wang and P. Dasgupta. Remote User Authentication Using VMM-based Security Manager[Online]. Available:http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.65.3633&rep=rep1&type=pdf. 2007, May 8.

[6] R. Gennaro, P. Karger, S. Matyas, M. Peyravian. Secure Key Recovery[Online]. Available:http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.119.4486&rep=rep1&type=pdf.

[7] D.E. Denning. The US Key Escrow Encryption Technology. *Computer Communications.* 1994, **17**(7): 453-457.

[8] Matt Blaze. Key Management in an Encrypting File System. *In Proceedings of the Summer 1994 USENIX Technical Conference*. Boston, MA, June 1994.

[9] Cylink, CyKey: Cylink's Key Recovery Solution. [Online]. Available: http://www.csm.ornl.gov/~dunigan/cykey.pdf, March 1997.

[10] K. Narimani and G. B. Agnew. Key Management and Mutual Authentication for Multiple Field Records. *Proc. of the Third International Conference on Information Technology: New Generations* (ITNG'06), 2006.

[11] Kazumasa Omote, Kazuhiko Kato. Protection and Recovery of Disk Encryption Key using Smart Cards. *In Proceedings of the Fifth International Conference on Information Technology: New Generations*. 2008, pp.106-111.

[12] Shamir. How to Share a Secret. *Communication of the ACM*. 1997, **22**(11): 612–613.

[13] Niels Ferguson. AES-CBC +Elephant diffuser: A Disk Encryption Algorithm for Windows Vista[Online]. Available:http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.1617&rep=rep1&type=pdf. 2006, August.

[14] BitLocker Drive Encryption Technical Overview [Online]. Available:http://technet.microsoft.com/en-us/library/cc732774(WS.10).aspx

[15] Paul Barham, Boris Dragovic, Keir Fraser. Xen and the Art of Virtualization. *In Proceedings of the 19th ACM Symposium on Operating Systems Principles*. 2003, pp. 164-177.

[16] Yevgeniy Dodis, Joel Spencer. On the (non)Universality of the One-Time Pad. *In Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*. 2002, pp. 376-385.

[17] Pratt, Fraser, Hand, Limpach, Warfield, Magenheimer, Nakajima, Mallick. Xen 3.0 and the Art of Virtualization. *In Proceedings of the Linux Symposium* . Ottawa. 2005, **2**: 65-77.