2009 Third International Conference on Emerging Security Information, Systems and Technologies

A new approach to protect the OS from off-line attacks using the smart card

Hossein Rezaei Ghaleh
Department of Electronic, Computer and IT
Islamic Azad University of Qazvin
Qazvin, Iran
hrezaei@qazviniau.ac.ir

Shahin Norouzi
Department of Information Technology
Ministry of Science, Research and Technology
Tehran, Iran
sh_norouzi@msrt.ir

Abstract— **Since the present computer systems use layered and modular architectures and execute the instructions in a number of different phases, therefore it has become an imperative to establish a trusted chain between various layers. It usually is integrity checking by hashing of executable codes. With guarantee of software integrity, the web servers and other network entities can trust to client systems or workstations. Several methods have been proposed for this purpose, each of them have their own advantages and weakness. Recently a group of big software and hardware companies working in Information Technology field known as Trusted Computing Group (TCG) are engaged in designing and making standards of various aspects of trusted computer systems including applications, PCs, networks, cryptography modules and so on. These standards can make the system trusted, but they need some hardware changes such as BIOS and Trusted Platform Module (TPM). These changes are not applicable for present systems and we have to purchase new hardware. This paper is an attempt at propose a new method that can make the present systems trusted. This method uses a removable trusted storage that is compatible with TCG storage standard.**

Keywords- OS security; off-line attack; smartcard; trusted computing;

I. INTRODUCTION

Security of information is an issue that has been taken into consideration form many years ago and by computer systems, this concept has gained further importance. The trusted computing, in its wider sense, comes under this concept and a group of big software and hardware companies working in Information Technology field known as Trusted Computing Group (TCG) are engaged in designing and making standards of various parts of trusted computer systems including applications, PCs, networks, cryptography modules and so on [23]. One topic which has gained the attention of the said group is secure computer startup. As these days the computer systems use layered and modular architecture and execute the instruction in a number of different phases, it is imperative to establish a trusted chain between the layers. For example, when we are not sure of the operating system's security and it being cleared from malicious codes, we can not trust the security of the application being executed by the system. The secure startup of the computer means that we make sure of the integrity of the each layer before its loading. Consequently, when the system is booted we would be certain that there exist no malicious code in it, or its data have not been tempered by unauthorized persons.

There are various methods for achieving this objective that each of them having their advantages and disadvantages which will be consider in this paper. Also in this paper, a method called PCSM has been proposed which in addition to having most of the advantages of the previous methods, has further parameters such as compatibility with present systems as well as more flexibility. Meanwhile, by making use of the idea suggested in this method, in return for losing a little performance, a number of major attacks to the said computer systems can be prevented.

In this paper, the earlier methods are briefly reviewed and then PCSM along with hardware and software architectures are explained. Next section is devoted to evaluation of various security analyses such as comparison of security goals. And finally, a summery of the scheme and the conclusion is provided.

II. RELATED WORKS

Many projects have been down on secure booting, application isolating and other technique for securing system with different approaches such as home and network applications. In this section most of the efforts made in this area are examined.

A. AEGIS

This method makes some modification on the standard startup of the computer and by adding a chip to the motherboard, uses this device as a root of trust. In this method, the digital signature has been employed and each layer before launching of the next layer verifies its digital signature. By doing this, upon loading of each layer, a chain of trust is formed. In case of any error during this process, the system automatically connects to a special server and recovers the lost module. The chip used in this method contains codes needed for cryptography, required network protocols for recovery of the lost module and the digital signature of the one or a number of lower layers of the system [9, 11 and 16].

B. sAEGIS

In this method some new capabilities have been added to AEGIS and some improvements have been made. One of these arrangements is using of smart card for access to the system. This card contains the digital signature of higher

978-0-7695-3668-2/09 $25.00 © 2009 IEEE
DOI 10.1109/SECURWARE.2009.57

layers such as operating system and applications and is protected by a PIN code [8].

C. U-Key Method

In this method, no modification is applied to the standard startup of the computer and the digital signature is generated for the higher layer only. Smart card is used for storing the digital signature and private key are also used in this method. Since the trusting point of this method is system's BIOS, therefore its security level is lower than the two above mentioned methods. But, this method is comparatively more flexible and it could be used in wide range of systems [5].

D. Trusted Linux Client

The goal of the Trusted Linux Client (TLC) project is to protect desktop and mobile Linux clients from on-line and off-line integrity attacks, while remaining transparent to the end user. This is accomplished with a combination of a Trusted Computing Group Trusted Platform Module (TPM) security chip, verification of extensible trust characteristics, including digital signatures, for all files, authenticated extended attributes for trusted storage of the resultant file security meta data, and a simple integrity oriented Mandatory Access Control (MAC) enforcement module. The resultant system defends against a wide range of attacks, with low performance overhead, and with high transparency to the end user [14, 15 and 21].

E. BitLocker

This method is a part of Windows Vista provided by Microsoft and uses TPM. This software in order to provide the highest level of security requires the TPM hardware to be installed on the motherboard and support special BIOS. The unique advantage of this software is that it provides appropriate protection for the user's data through encryption before storing the data on the hard disk and during the execution. It also provides separate partitions for the operating system and its modules and keeps them also as encrypted data and is also capable of activating the access control service [6].

F. VMM Based Methods

NGSCB [27] and Terra [28] are different methods that attempt to make the system secured by same idea. Both NGSCB and Terra explore a virtual machine monitor (VMM) to partition a tamper-resistant hardware platform into multiple isolated virtual machines. In NGSCB, a system is partitioned into two parts: trusted and untrusted, and only the trusted part is attested. Therefore, to ensure service trustworthiness, the service provider platform has to treat the service and all its code as trusted, which may not be true all the time. Terra partitions the system into virtual machines, each of which may be dedicated to a single application (e.g., a service). As such, the trustworthiness of a service can be evaluated by attesting its virtual machine. This attestation, however, is done at memory block level, which incurs high CPU and memory overhead. Terra achieves higher assurance of attestation because of strong process isolation provided by VMM, but lacks the capability of ensuring simple and efficient trusted execution across transactions.

III. PCSM

A. General Design

The proposed method in this paper consists of several security ideas. The first idea is that the required services for securing the system be placed in the layers lower than the operating system. By doing this, viruses, worms and other malicious programs, executed at the operating system layer can not crash or tamper security services. This idea is being currently used in many security computer systems [4, 7, 10, 12, 13, 18, 19 and 28] including in most of the Internet servers known as virtual servers. In this project a middleware is designed which uses the virtual machine for this purpose.

Another concern is creation of a secure environment for storing the user's private keys. In previously mentioned methods, in order to create such environment, TPM or a similar chip is used which should be installed on the motherboard and keys are defined corresponded to a specific machine. While in this project, the smart cart is used that from the level of security mechanisms and standards, is similar to that of TPM, but at the same time is portable and is defined corresponded to a user.

The other idea is that the entire of middleware as whole, to be stored as an image in disk and by the time of booting is loaded as Live-OS. In other words, a partition similar to that of the partition of the hard disk is created virtually on the main memory which is active while the system is on and exits in memory by the time the system is turned off [20].

For secure startup, modules such as boot loader and middleware image are stored in a specific partition of the disk which is write-protected and for any modification, the password authentication is required. By such arrangement, the integrity of the lower layers is guaranteed. We can store middleware on host hard drive and save boot loader on write-protected partition, but in this situation we have to verify middleware integrity at boot time and we have to develop smart card library, cryptography library and real-mode USB driver. When we use middleware, we have these libraries in OS level.

For protection of user's data, the data are encrypted for storage during the execution. This could be done in two ways, first by using encrypted file systems which use software for encryption and second by using hardware-based encryption [3, 24 and 25].

Another idea is putting all the hardware and software modules on a device with USB port which could be easily carried by the user. It will be elaborated more in coming sections.

B. Hardware Architecture

In this project all the hardware and software modules are put in a special device with schematic architecture of Figure 1. As it is shown, this hardware consists of three main parts including protected memory, secure token and USB hub. The memory has a special partition. In this partition, the writing access by password is controlled but there is no restriction

for reading. The specification of such memory has been standardized by TCG group [26]. The secure token is a card reader based on PC/SC standard which can hold a smart card with SIM format. Within this card reader, there is a contact smart card which supports ISO/IEC 7816 standard. The third part is a USB hub which provides the output of the two other parts as a single USB port. All these parts are gathered in a package and are connected to the computer via USB port [1, 2].

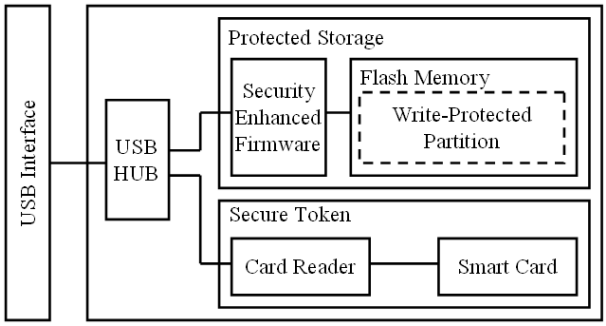

Figure 1. PCSM hardware architecture

C. Software Architecture

In this project, the software architecture is designed in a three layer form as shown in Figure 2. The lower layer shows the host's hardware. The Thin OS is a customized small size and high speed operating system. The Startup Services is responsible for managing and executing special boot process and providing the connection to the smart card. The CFS section is responsible for encrypting of the stored data of the upper layer on the relevant partition of the disk and supports AES standard. The Security Service section consist antivirus, firewall, intrusion detection and etc. The VM is responsible for virtualization the hardware for the upper layer and in fact is a computer which could be seen by the user. The upper layer is a software layer and is similar to usual computer system and has its own operating system, applications and data [1, 2].

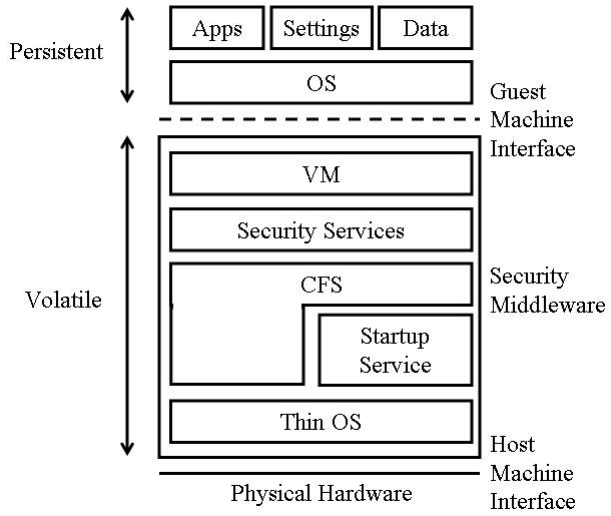

Figure 2. PCSM software architecture

D. Boot Process

The execution process in PCSM is that upon the turning the system on and completion of basic phases, the control is transferred from BIOS to boot loader installed on PCSM disk which has USB port.

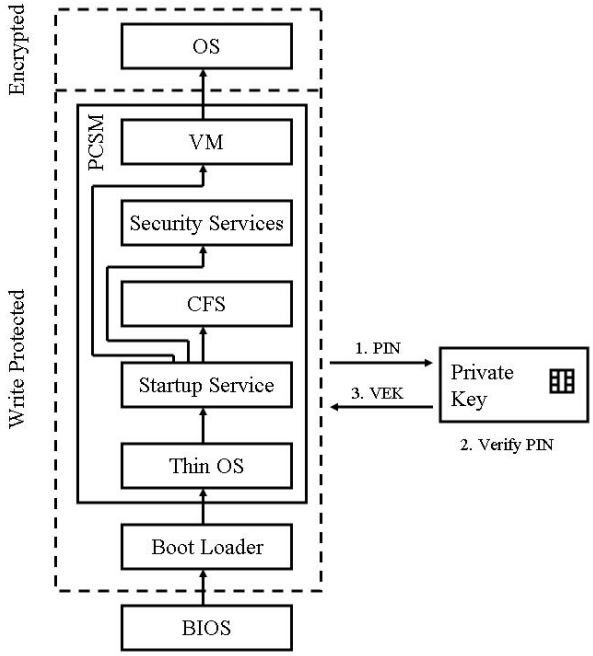

Figure 3. PCSM boot process

The boot loader which is written on the read-only memory loads the image of the middleware to the system's main memory and passes the control to it. The Thin OS connects to smart card and executes a program which receives the PIN code from the user. This code is then passed to the card and in case of its confirmation the Volume Encryption Key (VEK) is read from the card. Then this key is passed to the CFS to be used as encryption key.

At this stage the virtual machine which is loaded over the CFS, starts up. This machine loads the main operating system of the user and the normal process of loading of software starts. Main OS and user's data have been saved in an encrypted image file on host machine's hard drive. Encryption of user's data and the main operating system which are both located at the upper layer, is carried out during the execution process while hidden from the user. When the system is turned off, the Thin OS comes out of the main memory and the system is closed in a secure manner [1, 2]. This process is shown in Figure 3.

E. Implementation Details

In this project, from the general design, a sample is implemented. In this section, details of this implementation are discussed. In the said implemented sample, a device called SIM Reader Combo manufactured by *Eutron* has been used. This device has a 2 GB flash memory and the card reader of *CardMan* 6121 model made by *OmniKey* has been used.

In the boot loader section, the *GRUB* software which has a high degree of flexibility and customization has been used. For Thin OS a special version of Linux called *Slax Frodo* which has small size and is modular, has been used. In PCSM, a number of modules are added or taken away from the Thin OS. In the CFS section, the *TrueCrypt* software which contains the required algorithm of this project has been used.

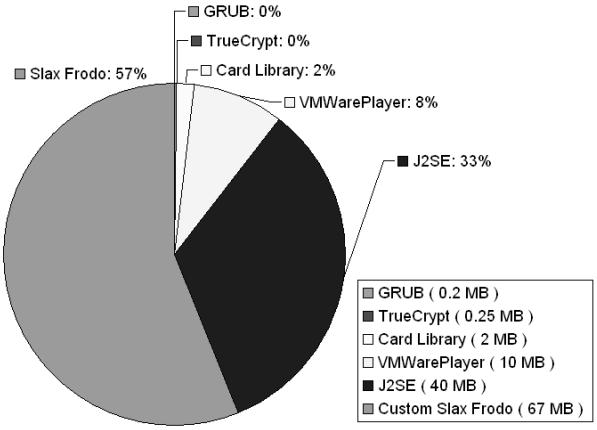

Figure 4. The used memory by various sections of the implemented PCSM

For programming environment and working with smart card, J2SE along with libraries such as JPCSC and card reader driver are used. The *VMWare* Player has also been used as virtual machine. The Figure 4 shows the size of the used memory by each software section in the implemented sample. The size of all components of the middleware is less than 120 MB.

In contrast to the other methods, due to the special feature of architecture of the PCSM, any kind of operating system could be used in the upper layer (user layer). In the implemented sample; the widely used operating system (Windows XP Professional) has been used [1, 2].

IV. Security Analysis

In this section, the security goals of this project are explained, and PCSM and other methods are considered for comparing. It is noticeable that, only off-line security attacks and their defensive methods are examined, so the on-line security attacks are not supported by these methods. In other word, when the OS protection and permission services are off, these attacks can be occurred.

Five main security goals that have been considered in this paper are firmware integrity guarantee, boot software integrity guarantee, OS components integrity guarantee, critical application integrity guarantee, and user's data privacy. They include active codes, which are executable and interpretable programs, and configuration files. The comparison among these goals is summarized in Table I.

According to the results, both AEGIS and sAEGIS are same and good in low-level software integrity guarantee like firmware, boot, and OS because they have the chip on motherboard and can access it every time. However, they do not guarantee high-level software like application programs and do not have enough flexibility for supporting them.

Methods using U-Key have only OS integrity and some application integrity programs guarantee because they do not have access to the chip on motherboard.

TLC, BitLocker, Terra, and NGSCB, which use TPM, have acceptable security level in the all of the five goals. They have the chip on motherboard and require software libraries for accessing in high-level software. However, they are not suitable for our purpose because they need some hardware changes, and they are not applicable in present systems. Moreover, both TLC and BitLocker are strong dependent to the hardware, and they only support specific OS.

PCSM has acceptable security level in all security goals except firmware integrity guarantee because it does not have any trusted chip on motherboard. PCSM can guarantee both its boot software and middleware integrity by write-protected partition on disk. In addition, when we use virtual encrypted volume for saving, OS and application integrity are guaranteed and also user's data privacy is supported because a key (VEK), which is stored in smart card securely, is required for reading or writing.

V. Some Security Attacks and Defenses

A. Phishing

In some of the attacks against the operating system, the attacker by cloning the User Interface of the operating system tries to retrieve the name and password of the user and gain access to confidential information. These types of attacks are called Phishing. For example, by replacing GINA module in the Windows, such attacks could be done. This type of attack was carried out by authors of this paper against Windows XP. By using PCSM this type of attacks could be prevented because the operating system along with all its modules are first encrypted and then saved and therefore it could not be replaced when the system is not on.

TABLE I. Comparison of Security Goals

Goals	Methods							
	AEGIS	***sAEGIS***	***U-Key***	***TLC***	***BitLocker***	***NGSCB***	***Terra***	***PCSM***
Firmware Integrity	Yes	Yes	No	Yes	Yes	Yes	Yes	No
Boot Integrity	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
OS Integrity	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Application Integrity	No	No	No	Yes	Yes	Yes	Yes	Yes
Data Privacy	No	No	No	Yes	Yes	Yes	Yes	Yes
No Hardware Change	No	No	Yes	No	No	No	No	Yes

B. Trojan Horse

The penetrating codes, known as Trojan horse are another attack to operating system services. In this method the user will not notice any unusual behavior but behind the scene, the user's data would be subject to unauthorized access. These destructive codes could affect drivers, services or the libraries of the operating system. By using the PCSM method this problem could be avoided because the modules of the operating system are first encrypted and then stored.

C. Dictionary Attack

Another attack carried out against the operating system is discovering the information about the user such as username and password by extracting their hash value in the account files of the operating system. This attack was also carried out by the authors of this paper against the Windows XP and it could be carried out on Linux as well. But PCSM, by encrypting the modules of the operating system including registry files prevents this type of attacks also known as dictionary attacks.

D. Logon Bypassing

Another concerning issue is unauthorized access to user's data outside the domain of the operating system. In this type of attacks, by booting the system by an operating system other than the original operating system, the access condition is ignored and user's files, stored on the disk are accessed. This attack has also been carried out by authors of this paper against the NTFS and Ext3 file systems. In PCSM method, by using encrypted files in layer lower than the operating system, this type of attacks could be prevented.

E. Brute Force

Another type of attack against computer system is called Brut Force. In this kind of attacks the attacker by entering all possible form of the user's password, discovers its correct value. This type of attack is usually launched by software. But in PCSM method such attack can not take place because the root key is stored in the smart card and protected by the user's password and this password is the same as the card's PIN code. In the designed smart card, the wrong PIN code could only be entered three times and then the code will be blocked and PUK would be demanded. PUK, due to its large size, could not be attacked by Brut Force.

F. Firmware Corruption

A more sophisticated attack by viruses is the one that creates invalid modifications in the firmware. In this type of attacks, the internal codes of the EEPROM of the BIOS or other hardware which are updatable, are tempered with and consequently some sections of the system comes under attacker's control or becomes useless. This attack was also carried out by the authors of this paper via the BIOS manufactured by Phoenix. PCSM is capable of preventing this attack because all hardware, from perspective of the upper layer of the operating system, is considered as virtual and the attacker has no access to the physical hardware.

VI. Performance Evaluation

We have done some performance tests for evaluating PCSM by applying them on native OS (Windows XP) and PCSM (with Windows XP for guest OS). Moreover, we employed PassMark™ Performance Test which is software with many tools for performance tests like CPU, memory, and disk test suits. The general system properties of our used test system are: Intel Pentium IV CPU with 3 GHz, 2048 KB cache size, 400 MB available RAM, and FAT32 file system. An important difference between native OS and virtual OS was the available RAM. In virtual OS, it was 128 MB because host OS needs separated memory. However this is configurable and we can increase the guest OS memory. The results are summarized in Figures 5, 6 and 7. We have tested three main performance parts include CPU, memory, and disk. Figure 5 shows comparison for CPU tests. In this test we have done different 8 tests like floating point math, finding prime number, and image rotation. In Figure 6 memory test results are shown, and we can see that they are near to each other, considerably. This test includes some parameters like allocating small block, reading cached and uncached data, and writing in memory. Last test suite is disk test which is shown in Figure 7. Disk is very important because usually it is bottleneck for total performance. This test includes both sequential reading and writing, and random searching for reading and writing operations.

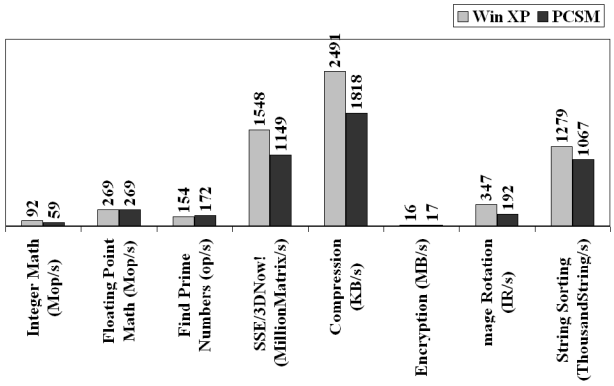

Figure 5. CPU test results

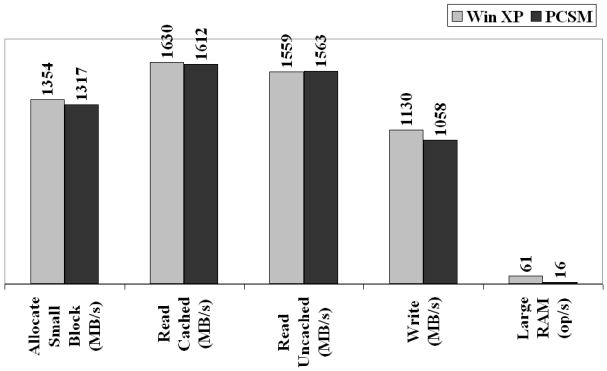

Figure 6. Memory test results

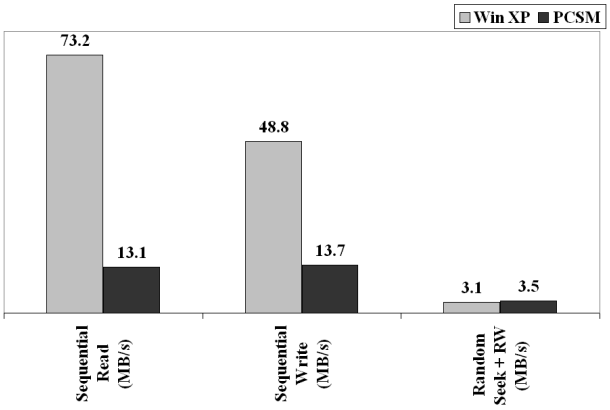

Figure 7. Disk test results

We can observe that CPU and memory have little difference in native and virtual OS. In other word virtualization has small and acceptable overhead. Furthermore, nowadays, virtual machines have 80 to 90 percent performance in comparison with native. On the other hand, we can see that the disk performance has more difference because PCSM uses virtualization and software-based encryption. Encryption is fulfilled by software, which its algorithm is AES, and all of data including user and OS data are encrypted. If we use hardware-based encryption, we will not have any encryption overhead in software layer. The performance in some existing disks with encryption in firmware is about 90 percent.

VII. CONCLUSION

This paper has reviewed the various introduced methods for securing client OS and then a new method called PCSM was proposed and was evaluated from security and performance perspective along with other methods. It became clear that this method is more flexible and has higher degree of compatibility with present systems. This method, due to its special architecture, is capable of preventing various security threats and therefore provides an acceptable level of security for personal computers and workstations connected to the network.

REFERENCES

[1] H. Rezaei Ghaleh, *PC Secure Bootstrapping*, M.Sc. Thesis, Islamic Azad University of Qazvin, March 2008.

[2] H. Rezaei Ghaleh, M.A. Doustari, *A new approach for secure and portable OS*, SECURWARE2008: The Second International Conference on Emerging Security Information, Systems and Technologies, IEEE Computer Society, August 2008.

[3] Michael Austin Halcrow, *eCryptfs: An Enterprise-class Cryptographic Filesystem for Linux*, International Business Machines Inc., 2005.

[4] Paul Barham, Boris Dragovic, and others, *Xen and the art of virtualization*, Proceedings of the nineteenth ACM symposium on Operating systems principles, 2003.

[5] Peng Shaunghe, Han Zhen, *Enhancing PC Security with a U-Key*, IEEE Security and Privacy, Volume 4, Issue 5, September 2006.

[6] *BitLocker Drive Encryption Technical Overview*, Microsoft TechNet, 2008.

[7] Peter M. Chen, Brian D. Noble, *When Virtual Is Better Than Real*, Proceedings of the Eighth Workshop on Hot Topics in Operating Systems, IEEE Computer Society , 2001.

[8] Naomaru Itoi, William A. Arbaugh, Samuela J. Pollack, Daniel M. Reeves, *Personal Secure Booting*, Proceedings of the 6th Australasian Conference on Information Security and Privacy, Springer-Verlag, 2001.

[9] William Albert Arbaugh, *Chaining layered integrity checks*, University of Pennsylvania, Philadelphia, PA, USA, 1999.

[10] Bernhard Kauer, *OSLO: Improving the security of Trusted Computing*, Technische Universität Dresden, Department of Computer Science, 2007.

[11] Arbaugh, Farber, Smith, *A secure and reliable bootstrap architecture*, IEEE Symposium on Security and Privacy, 1997.

[12] Hermann Hartig, *Security architectures revisited*, Proceedings of the 10th workshop on ACM SIGOPS European workshop, 2002.

[13] Daniela A. S. de Oliveira, Jedidiah R. Crandall, and others, *ExecRecorder: VM-based full-system replay for attack analysis and system recovery*, Proceedings of the 1st workshop on Architectural and system support for improving software dependability, ACM, 2006.

[14] Hiroshi Maruyama and others, *Linux with TCPA Integrity Measurement*, IBM Research, Tokyo Research Laboratory, January 28, 2003.

[15] Hiroshi Maruyama and others, *Trusted Platform on demand (TPod)*, IBM, February 1, 2004.

[16] William A. Arbaugh, Angelos D. Keromytis, David J. Farber, and Jonathan M. Smith, *Automated Recovery in a Secure Bootstrap Process*, Network and Distributed System Security Symposium, Internet Society, March 1998.

[17] James Hendricks, Leendert van Doorn, *Secure bootstrap is not enough: shoring up the trusted computing base*, Proceedings of the 11th workshop on ACM SIGOPS European workshop, 2004.

[18] Tal Garfinkel, Ben Pfaff, and others, *Terra: a virtual machine-based platform for trusted computing*, Proceedings of the nineteenth ACM symposium on Operating systems principles, 2003.

[19] Tal Garfinkel, Mendel Rosenblum, *When virtual is harder than real: security challenges in virtual machine based computing environments*, Proceedings of the 10th conference on Hot Topics in Operating Systems , IEEE, 2005.

[20] M. Nakamura, Seiji Munetoh, *Designing a trust chain for a thin client on a live Linux CD*, Proceedings of the 2007 ACM symposium on Applied computing, 2007.

[21] D. Safford and M. Zohar, *A Trusted Linux Client (TLC)*, Technical Paper, IBM Research, 2005.

[22] *TCG TPM Specification, Version 1.2*, Trusted Computing Group, 2005, https://www.trustedcomputinggroup.org/.

[23] *TCG Specification Architecture Overview, Revision 1.2*, Trusted Computing Group, April 2004, https://www.trustedcomputinggroup.org/.

[24] *TCG Storage Architecture Core Specification, Revision 0.9*, Trusted Computing Group, May 2007, https://www.trustedcomputinggroup.org/.

[25] Siani Pearson, *Trusted Computing Platforms: TCPA Technology in Context*, Prentice Hall PTR, 2002.

[26] David Challener, Kent Yoder, Ryan Catherman, David Safford, Leendert Van Doorn, *A Practical Guide to Trusted Computing*, IBM Press, 1 edition, January 6, 2008.

[27] Microsoft Corp. Next generation secure computing base, http://www.microsoft.com/resources/ngscb.

[28] T. Garnkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh. *Terra: A virtual machine-based platform for trusted computing*, In Proceedings of the 19th ACM Symposium on Operating Systems Principles, 2003.