A new approach for secure and portable OS

H. Rezaei Ghaleh
Islamic Azad University of Qazvin
hrezaei@qazviniau.ac.ir

M.A. Doustari
Shahed University of Tehran
doostari@shahed.ac.ir

Abstract

Data security has been an important concern from many years ago and has gained special importance in Information Technology. 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. This paper is an attempt at evaluation of these methods and proposes a new method called PCSM which tries to overcome the weaknesses of previous systems. This method is more flexible and transparent and with the proposed architecture can prevent many attacks and therefore provides higher level of security. This paper is concluded with a comparison between the proposed method and other methods.

1. 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 [22]. 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 portability of the secured data 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.

2. Related Works

Many projects have been down on secure booting of computers with different approaches such as home and network applications. In this section most of the efforts made in this area are examined.

2.1. 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

978-0-7695-3329-2/08 $25.00 © 2008 IEEE
DOI 10.1109/SECURWARE.2008.34

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 [8, 10 and 15].

2.2. 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 layers such as operating system and applications and is protected by a PIN code [7].

2.3. 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 tow above mentioned methods. But, this method is comparatively more flexible and it could be used in wide range of systems [4].

2.4. 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 [13, 14 and 20].

2.5. 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 too as encrypted data and is also capable of activating the access control service [5].

3. PCSM

3.1. 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, to be put 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 most of the Internet servers known as virtual servers. In this project a middleware is designed which uses the virtual machine for this purpose [3, 6, 9, 11, 12, 17 and 18].

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. In this project, the smart cart is used, which from the level of security mechanisms and standards, is similar to that of TPM and at the same time is portable and is defined corresponded to a user.

The other idea is that the 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 [19].

For secure startup, modules such as boot loader and middleware image are stored in a specific segment of the disk which is read-only and for any modification, the password authentication is required. By such arrangement, the integrity of the lower layers is guaranteed.

For protection of user's data, the data are encrypted for storage during the execution. This could be done in tow ways, first by using encrypted file systems which use software for encryption and second by using hardware-based encryption [2, 23 and 24].

Another idea is putting all the hardware and software modules on a device with USB port which could be easily carried by the user.

3.2. Hardware Architecture

In this project all the hardware and software modules are put in a special device with schematic architecture of figure 1.

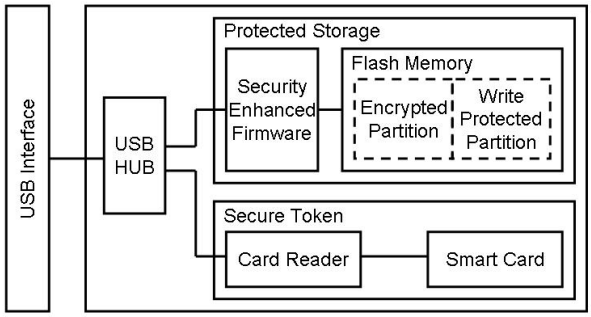

Figure 1: PCSM hardware architecture

As it is shown, this hardware consists of three main parts including protected memory, secure token and USB hub. The memory is divided into two partitions. In one of them, the writing access by password is controlled but there is no restriction for reading. The other partition holds the data in encrypted form. The specification of such memory has been standardized by TCG group [25]. 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 tow 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].

3.3. 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 bee 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].

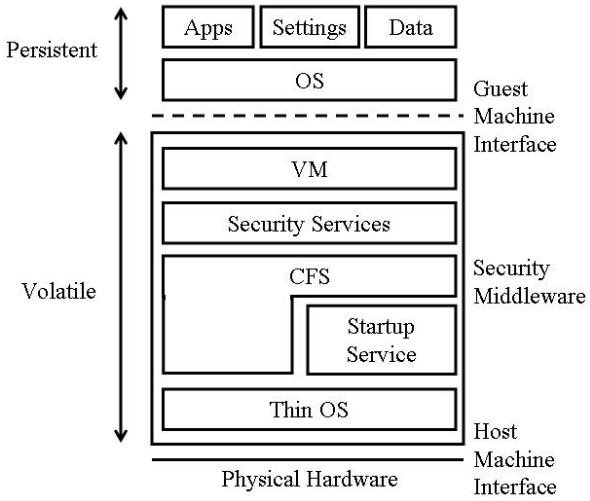

Figure 2: PCSM software architecture

3.4. Boot Process

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

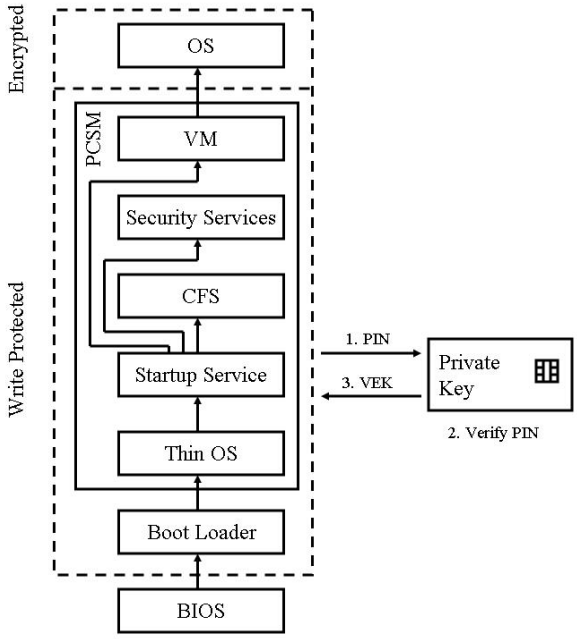

Figure 3: PCSM boot process

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.

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 and 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]. This process is shown in figure 3.

3.5. 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 made 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.

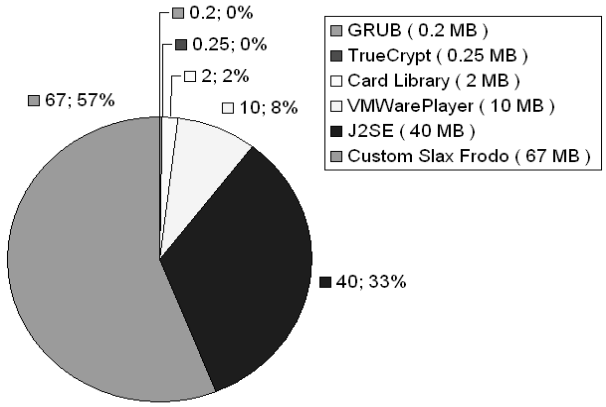

Figure 4: The used memory by various sections of the implemented PCSM

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].

4. Security Analysis

In this section we explain security goals of this project and consider PCSM and other methods and compare them. An important point is that we consider off-line security attacks and defenses and these methods do not support on-line attacks. In other word when these attacks have been done that OS is off. In this situation many attacks could occurred because OS protection and permission services is not running.

Five main security goals are firmware integrity guarantee, boot software integrity guarantee, OS components integrity guarantee, critical application integrity guarantee and user's data privacy. They contain active codes (executable and interpretable programs) and configuration files. The comparison of these goals has been summarized in table 1.

AEGIS and sAEGIS are same and both of them are good in low-level software integrity guarantee such as firmware, boot and OS because they need a chip on the motherboard and can access it every time. But they have not guaranteed high-level software such as applications and they have not enough flexibility for support it.

The method that uses the U-Key has OS integrity and some application integrity guarantee only, because it has not access to a chip on the motherboard. TLC and BitLocker that use TPM have acceptable security level in all of the five goals. They have a chip on the motherboard and required software libraries for access it in high-level software. But they are not suitable for our purpose for portable OS. They are very coupled with hardware and they support specific OS.

PCSM has acceptable security level in all security goals except firmware integrity guarantee because firmware is not fixed in different machines. Each machine has special hardware such as BIOS, motherboard, PCI cards and so on. PCSM can guarantee boot software and its middleware integrity by write-protected partition on disk. Also when we use virtual encrypted volume for saving, OS and application integrity has been guaranteed and user's data privacy is supported because a key (VEK) is needed for read or write and this key has been stored in smart card securely.

Table 1. Comparison of security goals

	AEGIS	sAEGIS	U-Key	TLC	BitLocker	PCSM
Firmware Integrity	Yes	Yes	No	Yes	Yes	No
Boot Integrity	Yes	Yes	No	Yes	Yes	Yes
OS Integrity	Yes	Yes	Yes	Yes	Yes	Yes
Application Integrity	No	No	No	Yes	Yes	Yes
Data Privacy	No	No	No	Yes	Yes	Yes

5. Performance Evaluation

We have done some performance tests for evaluating PCSM. These tests were done on legacy OS (Windows XP) and PCSM (with Windows XP for guest OS). Also we used PassMark™ Performance Test that is software with many tools for performance tests such as CPU, memory and disk test suits. Our test system had Intel Pentium IV CPU with 3 GHz, 2048 KB cache size, 400 MB available RAM and FAT32 file system. Tow important difference between legacy OS and virtual OS were that available RAM in virtual OS was 128 MB because host OS need separated memory, although this is configurable and we can increase the guest OS memory. The results have been summarized in figure 5. We have tested three main performance entity include CPU, memory and disk. Figure 5.A shows comparison for CPU tests. In this test we have done different 8 tests such as floating point math, finding prime number and image rotation. In figure 5.B memory test results have been shown and we can see that scores are very close. This test includes some parameters such as allocate small block, read cached and uncached data and write to memory. Last test suite is disk test that has been shown in figure 5.C. Disk is very important actor because usually it is bottleneck for total performance. This test includes sequential read, sequential write and random seek for read and write.

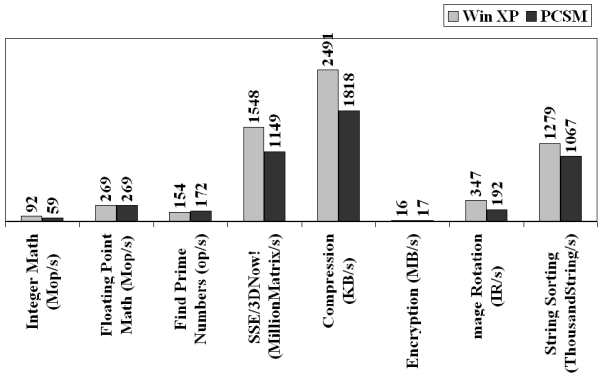

Figure 5.A: CPU test results

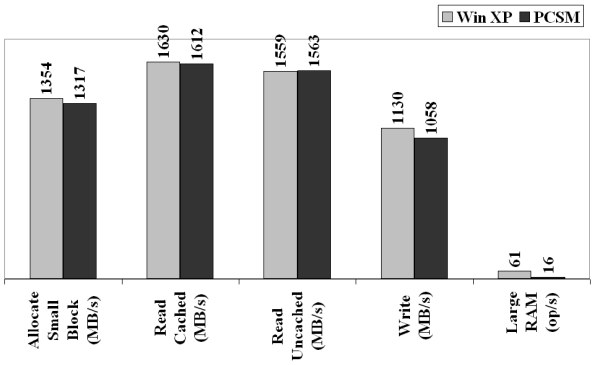

Figure 5.B: Memory test results

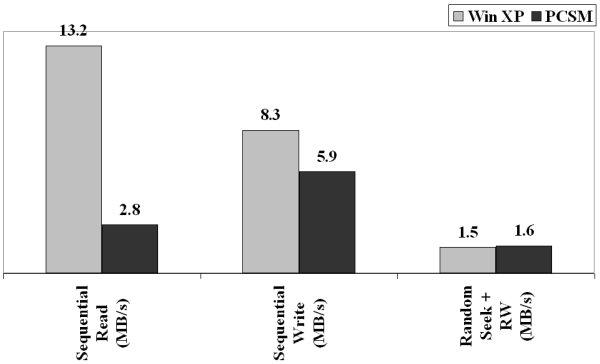

Figure 5.C: Disk test results

We see that CPU and memory have little difference in legacy and virtual OS. In other word virtualization has small and acceptable overhead. Also nowadays virtual machines have 80 to 90 percent performance in comparison with native.

But we see that the disk performance has more difference. The reason for this is that PCSM uses virtualization and software-based encryption. Encryption is performed by software, its algorithm is AES and all of data include user's data and OS are encrypted. If we use hardware-based encryption then we have not any encryption overhead in software layer. Some existing disks with encryption in firmware have efficiency about 90 percent.

6. 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 portability. 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.

7. References

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

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

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

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

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

[6] 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.

[7] 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.

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

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

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

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

[12] 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.

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

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

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

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

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

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

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

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

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