A Mobile Trusted Platform Module (mTPM) Architecture

Jesus Molina, Houcheng Lee, Sung Lee, Zhexuan Song
Fujitsu Laboratories of America, Inc.
8400 Baltimore Avenue, Suite 302
College Park, Maryland 20740, USA
{jesus.molina, houcheng.lee, sung.lee, zhexuan.song}@us.fujitsu.com

Abstract

We present the architecture of a mobile Trusted Platform Module (mTPM). Current implementations of TPM architectures assume physical bindings between the TPM and a single hardware platform. In this paper we introduce an alternative where a TPM is bound to a single virtual machine (VM) on a portable device, following the specifications provided by the Trusted Computing Group (TCG). By communicating with the VM infrastructure on the host machine, an mTPM offers various security functions similar as a physical TPM does. The mTPM is a mobile solution that will be beneficial to any professional who requires a portable, secure computing environment.

1 Introduction

The Trusted Computing Group (TCG) defines open standards [9, 10, 8], which provide the building blocks of hardware-based system security. The Trusted Platform Module (TPM), typically implemented in hardware, provides trusted information on the internal state of the system and stores cryptographic keys and identities. Security systems using the TPM are beginning to be deployed for applications which require enhanced data security levels such as electronic medical record systems.

Mobile users are users who must switch between computing devices, such as a doctor who must find the nearest terminal during emergency but unable to carry her own computing devices all the time. Although today, situation like that is considered limited, some researchers believe that in the near future, personal computing will become ubiquitous, in the sense that personal computing will be available anywhere on demand. Only when a user starts to use a computer, will it acquire her unique customization and states[5].

While systems based on TCG standards could offer a security solution, mobile users are yet to take full advantage of such systems. The bindings between the system and the TPM are physical (e.g., soldered to the PCI bus); thus, keys and cryptographic identities are stored locally in the TPM and possibly bound to the current state of the system. Whichever states mobile users leave in one computing device, stay with the device, and are not accessible when the same users want to resume their work on different platform. In order to overcome this limitation, the TCG provides migration features as part of the standard in order to import and export keys and identities if they are marked as *migratable*. However, the time involved in and the logistic problems related to acquiring the authorization to port between TPMs makes it unfeasible for jobs that require constant mobility between different platforms.

A naïve approach could be to just export the TPM functionality and APIs to an external mobile cryptographic coprocessor (e.g. smart card [3]). This external device might even have a secure storage, providing the solution for mobile keys and identities. However, more advanced features of TCG such as sealing and attestation would not be possible in such scenario.

In order to resolve these problems, we propose the mobile Trusted Platform Module (mTPM) architecture. The mTPM architecture employs a virtual machine along with the TPM in the form of a portable device. Virtual machines provide the necessary flexibility and portability in order to achieve our requirements while TPM offers the security solution. In our architecture, secrets are sealed into the portable computing environment, allowing the user to move the secrets and identities between different systems.

In this paper, we first provide a brief description of Trusted Computing technology, and overview of related technologies. In Section 3, we describe the architecture of mTPM, followed by a use case scenario in Section 4. We conclude this paper with the summary of prototype development in progress and future work.

2 Related Work

In this section we introduce the background for Trusted Computing and the previous work in portable environments.

2.1 Trusted Computing

The foundation of TCG is the TPM, a hardware *root-of-trust*. The most common implementation of a TPM is a chip physically attached to a computer. The TPM is accessed by software using a well defined command set. Through this command set, the TPM provides cryptographic functionality such as encrypting, signing, key generation and random number generation. It could also store a limited amount of information in non-volatile memory.

Additionally, the TPM contains a set of extensible Platform Configuration Registers (PCRs). As of the latest version, a total of 24 PCRs exist on the TPM. PCRs are used to store measurements on the current status of the platform. PCRs are reset when the system powers up and can only be reset or extended, but never directly modified. The extension of PCR is performed using a simple cryptographic algorithm: $Extend(PCR_N, value) = Hash(PCR_N||value)$. The $||$ sign stands for concatenating the two arrays, while "Hash" refers to applying cryptographic hash algorithm, in particular SHA1. The first measuring entity of the platform is trusted by default, as it is not previously measured by any other entities. This early measuring entity is called the *Core Root of Trust* (CRT) for measurement. For enhanced security, the CRT may be stored inside the TPM chip itself. Thereafter, all software entities launched are expected to continue the chain of trust by extending the PCRs before launching any other software. Each measurement is recorded and could be cryptographically verified using the PCRs by a verification party. The action of sending these measurements to a verification party for verification is called *attestation*. Hence, the TPM could be seen as a cryptographic repository for the measurements of the machine.

2.2 Portable Environment

One approach to achieve portability is the use of portable device as the storage for applications and related data. Several commercial products already provide portable environments, for example, U3 Smart Drives[12]. When a U3 device is plugged into a Windows based host machine, users can run the applications stored on the U3 device, and save the working state back to U3; this way, the users have the ability to access their own environment in any machine.

Another approach is the use of thin clients [11]. Thin clients carry a minimal amount of information, and they connect to a server to retrieve applications, user profiles, and data. Thin clients, however, should also

be secured, and the necessity of a central server limits the application of this solution to local environments (office, university).

In this area, a notable recent work is *SoulPad* [2]. The SoulPad architecture proposes dividing the computing environment into the static carcass formed by the hardware (denoted in the paper as *EnviroPCs*) and a mobile "soul". The mobile soul is implemented using a bootable USB hard disk, composed by an operating system to configure the underlining hardware, and a virtualized OS on top of it for flexibility to choose between operating systems. However, the only protection to the portable system is a password which decrypts the virtualized operating system, and no provision exist for identities or cryptographic measurements. As future work, they suggest possibly querying a TPM on the EnviroPC for correctness of the BIOS, but while this will provide with Trusted Computing Base (TCB) measurements, no sealing or key management operations will be possible. The architecture proposed in SoulPad could be used to add a mTPM to the soul, but due to performance (time to launch the environment) and usability (need a reboot of the PC) reasons we decided not to work directly under this architecture. Instead we use a Virtual Machine Monitor (VMM).

3 Architecture and Design

The objective of mTPM is to provide a secure, portable environment, while taking advantage of the TCG standards; in particular, mTPM utilizes a hardware-based TPM as its main security component. In this section, we describe the mTPM architecture.

3.1 Assumption

We refer to an *mTPM device* as a portable gadget that enables secure, portable computing, the *mTPM* as the TPM instance in the mTPM device, a *host* machine (or simply a host) as a computing device in which a mTPM device will be plugged, and a *host TPM* as the host TPM instance in the host.

When an mTPM device is plugged into a host, the host must securely launch the portable environment that is contained in the mTPM device. The mTPM in an mTPM device must implement full TPM functionality as defined by the TCG standards, allowing an mTPM to act transparently as a TPM. In addition, an mTPM must provide *attestation* capabilities that utilize the measured metrics from the host and adapt it to the mobile environment. An mTPM also provides a secure key and data storage to offer protection in the event of a lost mTPM.

In order to accomplish these goals, we assume that the following components exist on the host system:

1. A valid physical TPM based on the TCG standards and a TCG enabled BIOS and bootloader
2. A Virtual Machine Monitor (VMM) such as Xen [13] or sHype [6]

Note that the goal of mTPM is not to replace the host TPM, but to provide a personal TPM to the mobile user and utilize the host TPM. We believe that these two assumptions are reasonable. Many laptop and desktop computers are now being offered with TPM chips, and their use is rapidly increasing. The use of VMM (also called hypervisor) is exploding, and its use is now pervasive in distributed environments and servers. For instance, a stock operating system (OS) like Redhat Linux [4] includes the Xen hypervisor by default.

3.2 mTPM Architecture

The two main components of the architecture are the *host machine* and the *mTPM device*. The host machine features a VMM, a host TPM, and an external port in which the mTPM device can be plugged (e.g. USB).

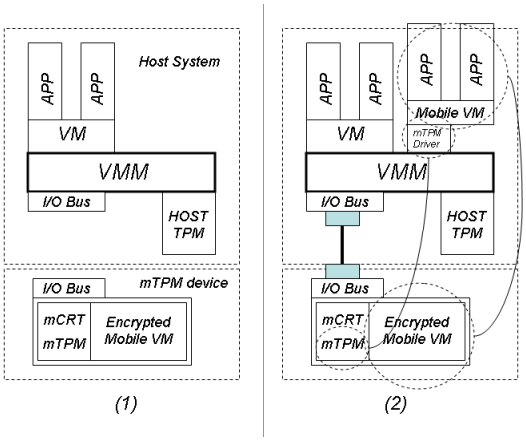


Figure 1: mTPM architecture components before (1) and after (2) connection

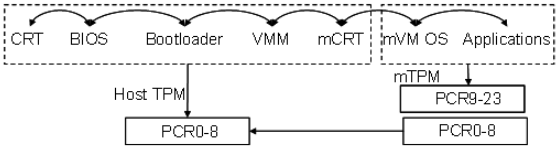
The mTPM device is composed of a *mobile Core Root of Trust* (mCRT), a *virtual machine* (VM) encrypted with a symmetric key, and an mTPM. The mCRT is a small program (possibly a bootloader or a microkernel) which is able to access both the mTPM and the host TPM. It also decrypts and boots the virtual machine contained in the mTPM device. As in the case of the CRT for the host, the mCRT could be stored inside the mTPM. The VM could be of any type, as long as it can be booted by the VMM at the host. The VM has been encrypted with a symmetric key, to avoid disclosure of data in the event that the mTPM device is lost or stolen. The mTPM fully implements all functionality defined by TCG standards including a shielded memory and a valid *Endorsement Key* from its manufacturer, which could be used to create attestation identities. Figure 1-(1) presents the components of the mTPM architecture.

The TCG is currently drafting a standard for optimizing TPM functionality on mobile devices [7]. We foresee an mTPM featuring the optimized mobile TPM and mobile devices to provide special functionality as mTPM devices.

The initialization process of mTPM is similar to that of the host TPM except that the ownership of mTPM must take place when it is plugged into a host with properly configured TPM. An mTPM also encrypts the VM contained on the same mTPM device during the initialization process. Similarly, the transfer of ownership process will require handling of the VM.

When an mTPM device is plugged into a host, the VMM measures the mCRT in PCR8 on the host TPM before launching it. As it is critical to ensure the integrity of the mCRT, we are actively looking into securing the mCRT and authenticating mCRT to the VMM. The mCRT then authenticates the user and decrypts the VM upon successful user authentication. Figure 1-(2) shows the components after this step. After the VM is launched, the chain of trust is maintained by the VM.

In Figure 2 we represent the measurement flow of the mTPM architecture and the expected final state of the PCRs. Any request for upper PCRs (> 8) should return the PCRs on the mTPM whereas the lower PCRs ($<= 8$) are values from the host.



mTPM PCR	Information	Physical TPM Present/Not Present
PCR0	CRT, POST BIOS,ROM	Inherited/0
PCR1	Motherboard Configuration	Inherited/0
PCR2	Option ROM	Inherited/0
PCR3	Option ROM data and conf.	Inherited/0
PCR4	IPL Code	Inherited/0
PCR5	IPL Data	Inherited/0
PCR6	State Transition	Inherited/0
PCR7	Bootloader	Inherited/0
PCR8	Hypervisor+mCRTs	Inherited/0
PCR9	VM Image	Measured/Measured
PCR10-24	Applications	Measured/Measured

Figure 2: Measurement Flow and PCRs End State

3.3 mTPM vs. vTPM

Creating a virtual TPM (vTPM) for servers with several virtual machines was discussed in [1]. In vTPM, a new virtual machine is proposed to act as a "TPM server" for all the virtual machines. This new virtual machine will launch a vTPM instance for each virtual machine in the server.

Our work defines the architecture for using a portable virtual machine linked to a physical but portable TPM, instead to a set of vTPM, which makes both the problem and the solution very different. The architecture coincides in how the PCRs are inherited, in order to achieve compatibility between the mTPM architecture and the vTPM architecture. However, we expect our solution to be used mainly in desktops instead of servers, the main target for vTPM. A matrix with the different solutions for physical and virtual TPMs is summarized on Figure 3.

		Computing Platform	
		Physical	Virtual
TPM	Physical	PC TPM Specs Target: All	mTPM Target: Mobile Desktop
	Virtual	TPM Emulator Target: Testing	vTPM Target: Servers

Figure 3: Solutions for Physical and Virtual TPMs

4 Application

In this section we will describe one scenario of using mobile phones as mTPM devices. Mobile phones are pervasive, and while their computing capacities are increasing, they still can only provide limited features. Moreover, even if the computing powers are greatly extended, the peripherals, input and output devices (screen, touchpad) will always need to be of reduced dimensions, decreasing usability.

The proposed mTPM architecture may be added as an additional feature for mobile phone users. This

way, the user has access to a portable secure environment, without having to rely on software existing on external machines. In fact, mobile phones are expected to ship in the near future with TPMs, so the hardware modifications needed for the mobile phone to work as an mTPM device will be minimal. The phones could be bundled with a mCRT and a mechanism to choose from two booting alternatives: As a TPM enabled phone or as a mTPM. The VM could be stored in the phone. As the capacity of the phone may be limited (although, capacity is comparable easier to increase by using external cards) the VM may rely in external web-based software to function. In a normal situation, the user will reboot the phone and choose, possibly with a switch, the mTPM mode. Then it will plug the phone to a machine, which will boot the internal VM, providing the user with a mobile secure environment, with all the provisions of the TCG only by carrying a phone.

5 Conclusion

In this paper, we propose an architecture that leverages the benefits of virtual machines with the benefits of trusted computing. Our architecture mitigates the security threats related to using mobile environments on an untrusted machine. We are in the process of developing a prototype that will demonstrate the feasibility of the proposed architecture. We have identified and tested out different components in mTPM. We expect to demonstrate full features of TPM including advanced features such as *Sealing*, *Migration*, and *Attestation*.

We expect the viability of this architecture will improve as TCG standards develop. This evolution will provide even stronger cryptographic algorithms and a variety of authentication mechanisms for the user. Hence, our proposed architecture does not rely on a fixed security solution, but on a security solution whose benefits will improve as the TCG standards mature.

References

[1] Stefan Berger, Ramon Caceres, Kenneth A. Goldman, Ronald Perez, Reiner Sailer, and Leendert van Doorn. vtpm: Virtualizing the trusted platform module. In *Security '06: Proceedings of the 16th USENIX Security Symposium*, pages 305–320, 2006.

[2] Ramon Caceres, Casey Carter, Chandra Narayanaswami, and Mandayam Raghunath. Reincarnating pcs with portable soulpads. In *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 65–78, New York, NY, USA, 2005. ACM Press.

[3] Helena Handschuh and Pascal Paillier. Smart card crypto-coprocessors for public-key cryptography. In Jean-Jacques Quisquater and Bruce Schneier, editors, *CARDIS*, volume 1820 of *Lecture Notes in Computer Science*, pages 372–379. Springer, 1998.

[4] Red Hat. inc. http://www.redhat.com.

[5] Mahadev Satyanarayanan. Seamless mobility on ubiquitous hardware. In *Keynote Speak, The 1st Annual International Conference on Mobile and Ubiquitous Systems*, 2004.

[6] Secure hypervisor. http://www.research.ibm.com/secure_systems_department/projects/hypervisor/.

[7] TCG mobile phone work group. https://www.trustedcomputinggroup.org/groups/mobile.

[8] TCG architecture overview. https://www.trustedcomputinggroup.org/groups/TCG_1_0_Architecture_-Overview.pdf.

[9] TCG TPM specification - design principles. https://www.trustedcomputinggroup.org/specs/TPM/-Main_Part1_Rev94.zip.

[10] TCG TPM specification - structures of the tpm. https://www.trustedcomputinggroup.org/specs/TPM/-Main_Part2_Rev94.zip.

[11] Thin client. http://en.wikipedia.org/wiki/Thin_client.

[12] U3 smart drives. http://www.u3.com/.

[13] Xen virtual machine monitor. http://www.cl.cam.ac.uk/Research/SRG/netos/xen/.