TeeMo: A Generic Trusted Execution Framework for Mobile Devices

Young-Ho Kim', Yun-Kyung Lee', Jeong-Nye Kim'

'Electronics and Telecommunications Research Institute
128 Gajeong-ro Yuseong-gu Daejeon, 305-700, Korea
{wtowto, neohappy, jnkim} @etri.re.kr

Abstract. Compared to the security solutions deployed in conventional personal computers, mobile devices have limited protection measures against the potential threats due to lack of computing resources and higher chances of loss or theft. In this paper we present TeeMo, a generic trusted execution framework that allows secure services to execute in isolation from the general applications running in the other domain. For this framework, domain separation is achieved by the virtualization technology. Our approach ultimately helps security-demanding applications get secure services inside the protected domain via inter-domain communications and also helps the TCB remain small.

Keywords: Mobile Security, Trusted Execution, Virtualization

1 Introduction

Recently smart electronic devices such as smartphones and tablets have been promising in the personal device market due to their high mobility and usability. With the easy accessibility of Wi-Fi, and cellular network to the Internet, the risk of damage from compromised devices has become much higher than ever expected. In addition, those services are requiring smart electronic devices to have trusted environments for software execution and sensitive information that should be protected against malicious attacks. Compared to existing security solutions deployed in personal computer, mobile devices have constrained computing resources: CPU, memory, and power consumption. Furthermore, mobile devices are highly likely to be stolen and missing that it is urgent to come up with technical measures to those threats. There have been prior studies to address the problems above. As in the case of PCs and server platforms, anti-virus solutions have been suggested but turned out to be ineffective. This paper proposes a practical and efficacious framework based on the virtualization technology, called TeeMo, which aims to protect important computing resources from malicious attacks. We designed a generic execution environment framework and interfaces for secure mobile devices.

The rest of the paper is organized as follows. Section 2 describes the threat model for our study. In section 3, we cover several design issues and solutions for providing mobile devices with trusted execution environment. In Section 4 we conclude and summarize our results.

2 Threat Model

Android has a robust security framework at an operating system level through the Linux kernel, a sandbox execution model, and a code signing mechanism for application to identify the author of the application and to validate its integrity. Android application runs in such a secure runtime environment in which it executes as a separate process under unique user identifier (UID) with its corresponding designated permission. Thus, programs complying with this rule can not infringe on other program's territory such as code and data section during the application's execution lifetime. As in the case of a desktop computer, root permission enables a user to have full access to all applications and applications' data. Android devices allow users to install a rooted operating system or applications that help users root their own devices with no difficulty. Accordingly granting the root permission by device owners would increase the vulnerability of the mobile device and expose their devices to the malicious attackers. And to make things worse, latest android malware variants can make a rooting attack on the remote devices intentionally, which uses a exploit code to gain root permission on the android smartphones.

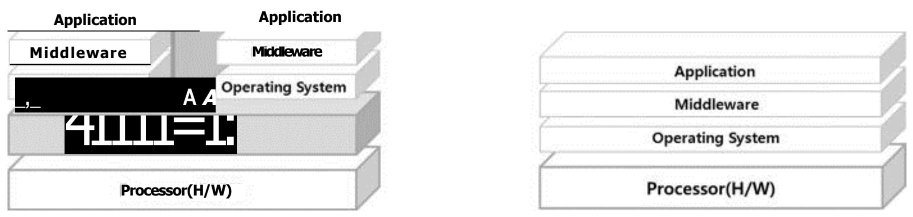
Linux kernel is a general purpose operating system which is based upon open source software project. The benefit of developing open source project is that various developers can be involved in the development and verification. Even several software driven by open source community have some bugs, let alone vendors' own device driver codes that are usually released with no public verification. Linux kernel is the most important and privileged component in android platform. Malicious attackers can exploit some bugs in kernel code which has basic security mechanisms to enforce security policy. Therefore, a bug in kernel code could lead to a meltdown of entire security measures deployed in Linux. Device drivers handling peripherals such as keypad, screen, camera could be compromised and used in data breaches. The more lines of code exist in the system, the worse the system can guarantee its integrity. From the result by the component based analysis, badly-written device drivers caused large amount of internal defects in Linux kernel. Given the plethora of device drivers implemented by many vendors, the possibility of vulnerability by kernel bug would be much higher than expected.

3 Design Issues

This section discusses the design issues and important properties that we consider to meet the security requirements for secure mobile devices. Our design is based upon three fundamental requirements:

3.1 Domain Separation

The domain is a distinct and standalone execution environment for a mobile device. In this paper, we assume that a domain simply means a single software execution environment that includes operating system, middleware, and application as a whole.



(a) **(b)**

Fig. 1. Computing system based upon (a) single domain and (b) multiple domains

As the left one depicted on Fig. 1(a), a general computing system is composed of the hardware, operating system, optionally middleware, and application as a service. For system described on Fig. 1(b), domain-isolated computing system would be comprised of hardware, necessarily virtualization, and a separated software platform running on each virtual machine created by the virtualization mechanism. In such a framework [9], there is believed to a virtual barrier between the domains, which can not be penetrable from both sides. However from the user perspective, the platform on a virtualized domain can be seen as a single platform on top of the real hardware.

Even under a compromised application or operating system, we can maintain the system's data integrity and execution privacy through the domain separation. Between isolated domains, it is strictly prohibited to make a direct call to the other domain's procedure or function. In this environment, security breaches caused by internal software loopholes or external attacks in one of the isolated domains does not affect the other domain's integrity. Also, each domain has no information about other domain such as operating system and applications, let alone internal API or security vulnerabilities. As such, virtualization can provide security by isolating critical components into separate protected domains within a single physical machine. In the framework we are proposing, a physical mobile system is virtually divided into two separate domains, wherein one domain is publicly exposed as a conventional platform and the other is critical domain hidden from other domains or even device owners. While the domain separation can be implemented by a software hypervisor [1] or a hardware supported solution[3], the type of implementation we consider is not limited to a specific technology.

3.2 Small TCB

The Trusted Computing Base (TCB) of a computing system is a set of components including hardware, firmware, and software that are crucial to its security. Therefore, a compromise of any component in the TCB could lead to collapsing the entire protection mechanisms within the system. The smaller the size of TCB is, the better the security of the system would be to validate the TCB. For this reason, TCB for a system is reduced to the combination of the hardware processor and the hypervisor software. Commodity operating systems such as Linux are huge in terms of LOC and usually are implemented in a monolithic architecture. Such a large aggregate and monolithic TCB [2] for general purpose operating systems is not adequate for trusted execution environment due to their kaleidoscopic and complex characteristics. There have been several prior studies to reduce TCB for secure software execution in a virtualization platform [4]. In this paper, we are assuming that there is no hardware and software flaw in the virtualization implementation. Secure service for security-demanding application is provided as a form of TCB component. We present TeeMo, a trusted execution framework for mobile device, which supports security services running in an isolated partition. To make security services to be a part of TCB, the isolated software platform for secure services should be small and easy to verify. Therefore, the software platform for secure services should be written in a small code size so that it will also help reduce the response of a security service between the domains

3.3 **Inter-domain Communication**

Most virtualization studies have been focusing on the isolation of processor, memory, and I/O devices efficiently and stringently. Software Smartcard [6], a software-based implementation of smartcard on the application processor acts as a SIM card and presents an isolated and secure environment for the cryptographic keys. Likewise, standalone security services running in the virtually sealed domain can benefit this strictly isolated architecture. However, this solution itself would not improve on the security of the public domain's conventional platform running on the application processor. In contrast, TeeMo presents a framework that facilitates the security services in the protected domain to help enhance the security and robustness of the application running in the public domain.

TCP/IP is a common protocol between two communication end-points in a single domain or in the distributed environment. In virtualized environment, XenSocket points out that the performance of inter-domain communication is very poor, compared to that of a UNIX domain socket on a native Linux system [8]. The authors of XWAY speculate that the poor performance mainly contributes to TCP/IP processing cost in each domain and long communication path between both sides of a socket [5].

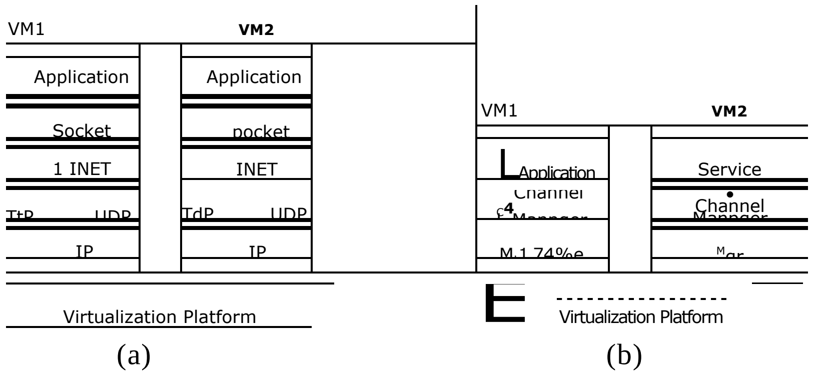


Fig. 2. Inter-domain communication based up (a) TCP/IP and (b) lightweight message pipe

Therefore, we design a lightweight message pipe as an inter-domain communication protocol rather than using conventional TCP/IP protocol stack. That decision is not only due to performance overhead incurred by TCP/IP but also to maintain the protected domain as a small TCB. In addition, in line with the end-to-end argument design principal [7] it is rational to move related functions upwards in a layered system, closer to the entities that use the function since security functions are placed at the highest level of a system in the protected domain.

4 Conclusion

In this paper, we have presented TeeMo, a generic trusted execution framework that provides security-demanding applications with secure services inside the isolated and protected domain while maintaining its TCB size small enough to be easily evaluated and validated. And we also have discussed the approaches we take to aim to address the threats and eventually to come up with the architecture of TeeMo in a virtualized environment. In our future work, we will implement TeeMo on a real virtualization platform and integrate it with the existing applications and services to evaluate our approach's effect on those applications from the security perspectives.

References

1. Jeremy Andrus, Christoffer Da11, Alexander Van't Hof, Oren Laadan, and Jason Nieh, Cells: A Virtual Mobile Smartphone Architecture. In: Proceedings of the 23th ACM Symposium on Operating Systems Principles (2011)
2. Patrick Co1p, Mihir Nanavati, Jun Zhu, William Aiello, George Coker, Tim Deegan, Peter Loscocco, Andrew Waffleld, Breaking Up is Hard to Do: Security and Functionality in a Commodity Hypervisor. In: Proceedings of 23rd ACM Symposium on Operating Systems Principles (2011)
3. Torsten Frenzel, Adam Lackorzynski, Alexander Warg and Hermann Hartig, ARM TrustZone as a Virtualization Technique in Embedded Systems. In: 12th Real-Time Linux Workshop (2010)
4. M. Hohmuth, M. Peter, H. Hartig, J. S. Shapiro, Reducing TCB size by using untrusted components — small kernels versus virtual-machine monitors. In: Proceedings of the 1 1 th ACM SIGOPS European Workshop (2004)
5. K. Kim, C. Kim, S. Jung H. Shin and J. Kim, Inter-domain socket communications supporting high performance and full binary compatibility on Xen. In: Proceedings of the fourth ACM SIGPLAN/SIGOPS, pp. 11--20 (2008)
6. Matthias Lange, Steffen Liebergeld, Adam Lackorzynski, Alexander Warg, Michael Peter, L4Android: A Generic Operating System Framework for Secure Smartphones. In: Workshop on Security and Privacy in Smartphones and Mobile Devices (2011)
7. J. Saltzer, D. Reed and D. Clark, End-to-end arguments in system design. In: ACM Transactions on Computer Systems, vol. 2, pp. 277--288 (1984)
8. Xiaolan Zhang, Suzanne McIntosh, Pankaj Rohatgi, John Linwood Griffin, XenSocket: a high-throughput interdomain transport for virtual machines. In: Proceedings of the ACM/IFIP/USENIX 2007 International Conference on Middleware, pp. 184--203 (2007)
9. Tal Garfinkel, Ben Pfaff, Jim Chow, Mendel Rosenblum, Dan Boneh, Terra: a virtual machine-based platform for trusted computing. In: Proceedings of the 9th ACM Symposium on Operating Systems Principles, pp. 193--206 (2003)