ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Indian Journal of Science and Technology, *Vol 9(24), 10.17485/ijst/2016/v9i24/96083, June 2016*

Research and Implementation of Type-1-Based Virtualization Security System in Smart Devices Environment

Ki-Bong Kim[1], Yong-Ho Kang[2], Chin-Hoon Kim[2] and Chang-Bok Jang[2]*

[1]Department of Computer Information Processing, Daejeon Health Institute of Technology, Korea; kbkim@hit.ac.kr
[2]R2Soft Corporation, Korea; kang@r2soft.co.kr, jhkim1@r2soft.co.kr, chbjang@r2soft.co.kr

Abstract

Background: The mobile virtualization security system can provide enhanced security with separated environment of simultaneously and independently operating multiple OS on the Hypervisor Abstraction Layer. **Methods Analysis:** This paper introduces the research and implementation of Type-1-Based Virtualization Security System in Smart Device Environment. Its detailed functions are mobile virtualization management, security authentication, security policy and access control, encryption/decryption, and safe storage that provide secure communication among the guest Operating Systems. This paper also introduces an experimental product operating a mobile office application. **Findings:** In the comparative test with the existing TEEMO Type-2-based Virtualization Security System, it measured processing times of compulsory loading situation and no-loading situation for 12 major security APIs. As a result, our system has showed reduced API processing time from 48% to 85%, faster processing speed in both compulsory loading situation and no-loading situation. Its average processing time has been 0.189 in the situation of consecutive API callings. Therefore, our system has proved acceptable processing time for real users as well as enhanced security with authentication and safe storage in the mobile hypervisor virtualization system. **Application/Improvements:** This research can offer assistance in providing secure communication and speed improvement among guest OS in mobile virtualization environment.

Keywords: Hypervisor, Mobile Security, Mobile Virtualization, Mobile Office, Smartwork

1. Introduction

Mobile devices continue to grow in prominence as they are increasingly utilized in business. They give strength that necessary works can be done with ease and convenience everywhere and anytime. Nevertheless, the advantage involves security-related problems including loss and burglary, abnormal wireless intrusion, unauthorized access to data, installation of malicious applications in mobile devices where various personal and business information are stored. For the effort of preventing such problems, enterprises are putting emphasis on mobile security. For mobile devices including smartphones, safe and ready data management system is required through integrated security so that only authorized users can access to personal and business information, important data are classified and saved in a safe storage, data are encrypted in case of accidents, authority is set up to access to critical data, and various authentications are provided.

In order to provide security and to cope with external threats, various types of security systems have been developed. Representative solutions are Wireless Network Access Control (WNAC) and Wireless Intrusion Prevention System (WIPS) which blocks and controls

*Author for correspondence

access with wireless devices, installed by unauthorized access point illegally. Companies or organizations with such security solutions remove all the open access points in order to prevent wireless intrusion from outside so that it is possible to protect external intrusions fundamentally[1-5].

The Mobile Device Management (MDM) solution provides a system to control mobile devices in a remote place at any time using the Over-The-Air (OTA). It is an integrated solution for mobile devices with management and collection of related information, trace management in case of loss/burglary, usage limitation for adoption of security policy, and distribution of software patch programs. With the MDM solution, data can be deleted to prevent information leakage if a mobile device is lost, e.g., with the initiation function of factory data[1-5].

With the advent of security systems, a broad part of business work can be carried out not only with personal computers but also with mobile devices. Documents which were shared inside of a company can be openly circulated outside. Now, protecting business contents as well as securing mobile devices are more essential than ever. Even though existing solutions let users to directly access important data from a user's OS, they don't provide an enhanced security OS which prevents information leakage. In order to resolve these challenges, various solutions have been introduced, and virtualization technology is a new trend of research in security.

2. Related Researches

2.1 Virtualization Technology

Virtualization technology provides a guest OS with virtualized devices by concealing physical hardware through software abstraction layer called Virtual Machine Monitor (VMM) or Hypervisor. General platforms have vertical hierarchy structure of hardware, operating system, and user applications, whereas hypervisor platforms can have multiple operating systems horizontally using virtualized various devices like the Figure 1. Therefore, different operating systems can run independently and simultaneously[6-11].

The primary reason to use a virtualization technology has been to maximize the use rate of hardware in a single device with high arithmetic function. Nowadays, another characteristic is being researched that the software which operates in an independent domain can provide reliability and security because access among operating systems are not allowed, as the domain of each operating system is independent in a virtualization environment. For this reason, virtualization can be utilized as a new way of security technology[6,7].

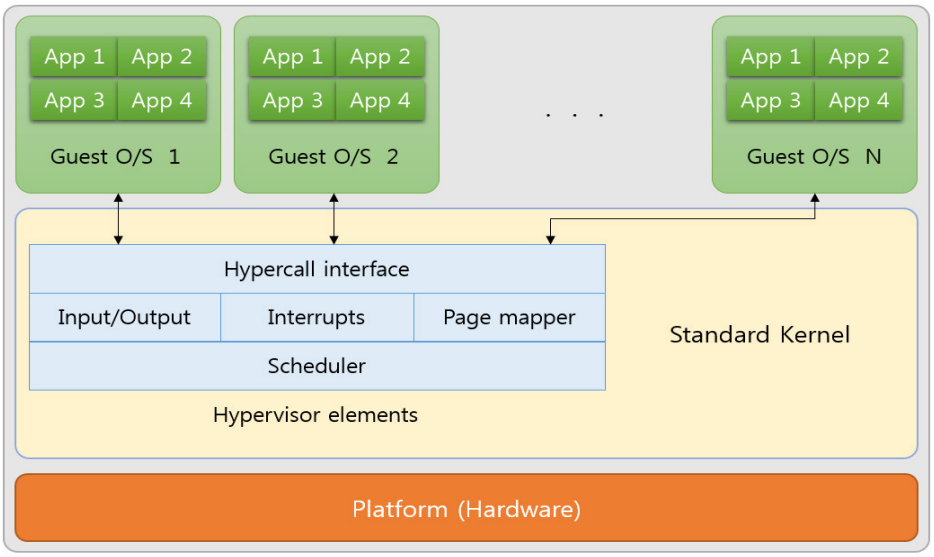

Figure 1. Hypervisor Virtualization Platform.

2.2 TEEMO Technology

TEEMO is a virtualization-based secure execution environment for mobile platform proposed by the Electronics and Telecommunications Research Institute (ETRI) of Korea. It aims to provide different security policies in separated two domains and to offer security service in a virtual machine isolated with a safe method. Various security services are offered in a secure domain, and users are allowed to use these services only through security service APIs. This technology uses Type-2-based hypervisor to separate the two domains. Generally, Type-2-based virtualization technology virtualizes with an emulator method, and its weakness is a slow processing speed[2,12].

2.3 TrustZone Technology

TrustZone is a virtualization security technology developed by the ARM. It provides a secure execution environment that two different domains exist in one CPU, not using a separated security hardware chip. As shown in the Figure 2, the basic operating structure of the TrustZone is that there are TZAPI and communication interface for client application to use security environ-

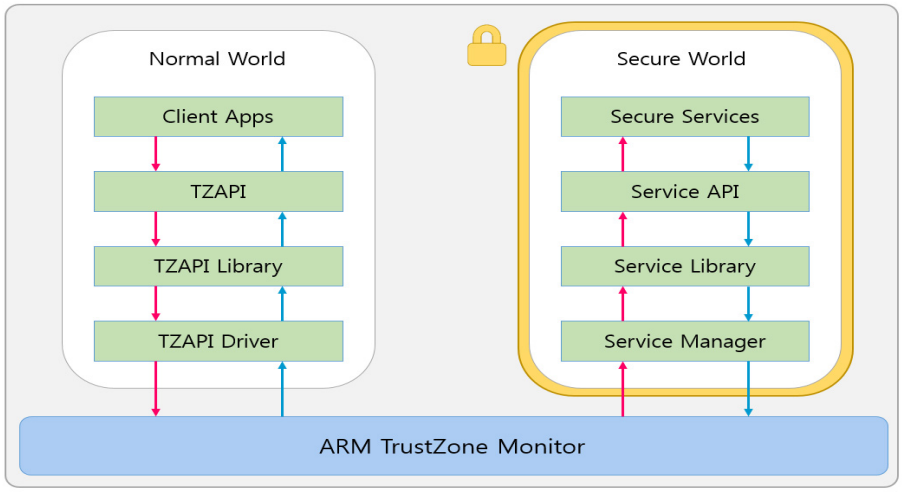

Figure 2. ARM TrustZone Message Transmission Architecture.

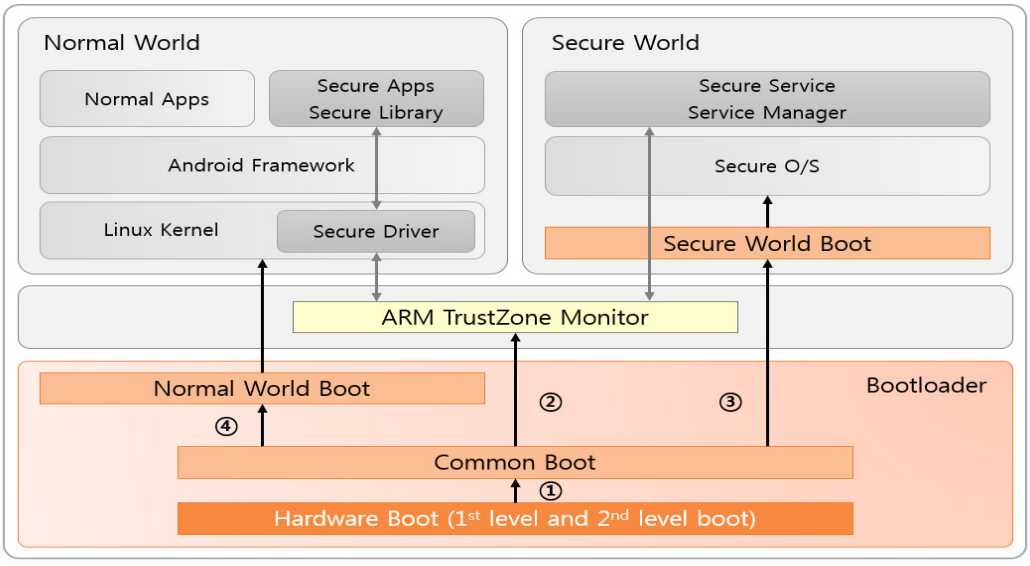

Figure 3. ARM TrustZone Booting Process.

ment and security service. The TrustZone Monitor can carry out message transmission as software abstraction layer which connects both domains. Figure 3 shows the TrustZone-adopted booting procedure in Android environment[13-15].

2.4 Xen on ARM

The Xen on ARM is a Xen-based mobile virtualization solution introduced by Samsung Electronics, Co., Ltd. It suggests various security technologies including Xen-based Secure Execution, Secure Boot, Secure Storage in order to enhance security of the platform. Its strength is that it doesn't need to implement new device drivers because it controls physical devices not in hypervisor but in Dom0, contrary to existing embedded virtualization technologies. But it has a big performance overhead. As DomU doesn't have the authority to access to physical hardware directly, it has a network and block driver. Hypervisor manages IPC process and scheduling among guest operating systems[16,17].

3. Type-1-based Virtualization Security System Structure

As suggested in this paper, Type-1-based virtualization security system for smart devices environment is composed of Normal Domain, Secure Domain, Mobile Hypervisor, and a mobile office application. The normal domain to which users are accessible is based on Android OS. The secure domain, where approved access is allowed only, is based on the uCOS with security function. The mobile hypervisor, based on Type-1, connects the two domains for their communication, and a mobile office application runs on the system.

The system users can install or run applications, which don't require special security, in the normal domain of the Android OS without restriction, same as in the existing environment. If the user wants to carry out a security-required work such as personal information or business task management, he/she accesses to the security service provided by the uCOS-based secure domain through authentication. The OS of the normal domain and the secure domain are separated each other. Access to the secure domain is possible through mobile hypervisor only so that malicious applications cannot be installed and inaccessible in the secure domain, and important data in the secure domain are prevented to be leaked.

Figure 4 shows the structure of the Type-1-based virtualization security system in smart devices environment. The physical hardware including CPU, memory, storage, I/O devices is virtualized by the mobile hypervisor and accessible like real hardware of the Android and the uCOS. Functions or applications of each domain are sep-

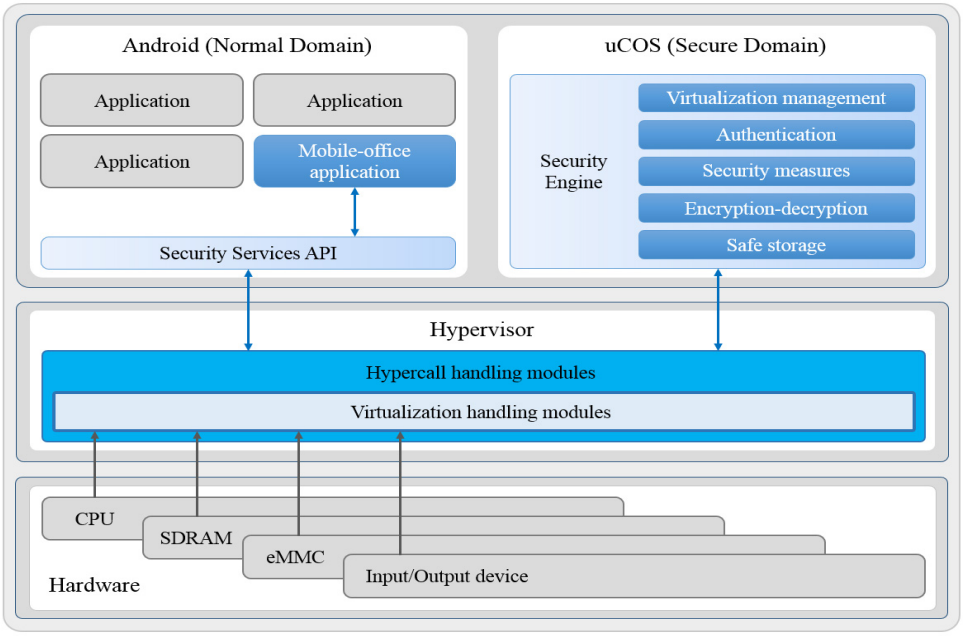

Figure 4. Type-1-Based Virtualization Security System in Smart Devices Environment.

arated to be accessible in the relevant domain only. The two domains are connected by the mobile hypervisor so that they can transmit information only after authentication. After that, the approved applications of the normal domain use security service APIs to access the secure service module which is provided in the secure domain.

This system utilizes the following development environmental factors to implement and test: The Ubuntu 11.10 server is used to make and compile security engine and service source code, and to load them into a targeting board. The Arm-2010q1-188 is adopted as a compiler. The Odroid-Q with the Exynos4412 CPU is used for a smart device as a targeting board. The mobile office application is developed under the JDK 1.6.0, Android API level 8, and Windows 7. Apache-Tomcat 8.0.28 is used as a web server for data connection, and the MySQL 5.7.9.1 is used as a database. For debugging, the Odroid Debug Board and the Teraterm are used. Additionally, the main functions suggested in this paper were implemented and tested using the security service API and the engine API processing security service, with technology transfer from the ETRI[2].

3.1 Implementation of Security Service and Security Engine Functions

In order to activate mobile virtualization functions in the normal domain and the secure domain by hypervisor, it is necessary to open and close channels and sessions for the connection of the two domains by communicating with hypervisor in each domain. Once safe channel and session are open, it is implemented to have the same effect as calling a procedure of service which is provided inside, when using security functions in the normal domain.

In the virtualization management module, processes including session management, procedure calling dynamic management, and message format management are performed in order to safely provide all the security functions for applications inside of the normal service domain. Figure 5 shows a channel start message printed from a secure engine when a mobile virtualization channel is open.

Authentication is a key function that only authorized users can access to the OS of the secure domain. It confirms that users or applications of the normal domain are authorized, and creates access control authority. When security service APIs in the normal domain are requested by a user to access to a security module in the secure domain, it transmits authentication information to the authentication module in the secure engine. If authentication is successfully carried out based on the transmitted information, the authentication module creates an authentication token and passes it to the access control module thereby approving access to the secure domain.

In addition, in case it is allowed to access to the secure domain with successful authentication, security policy functions are needed to provide differentiated access control authorities in accordance with the characteristics of the secure function or the importance of data. Security policy functions are categorized as security policy management, access authority decision, and access control

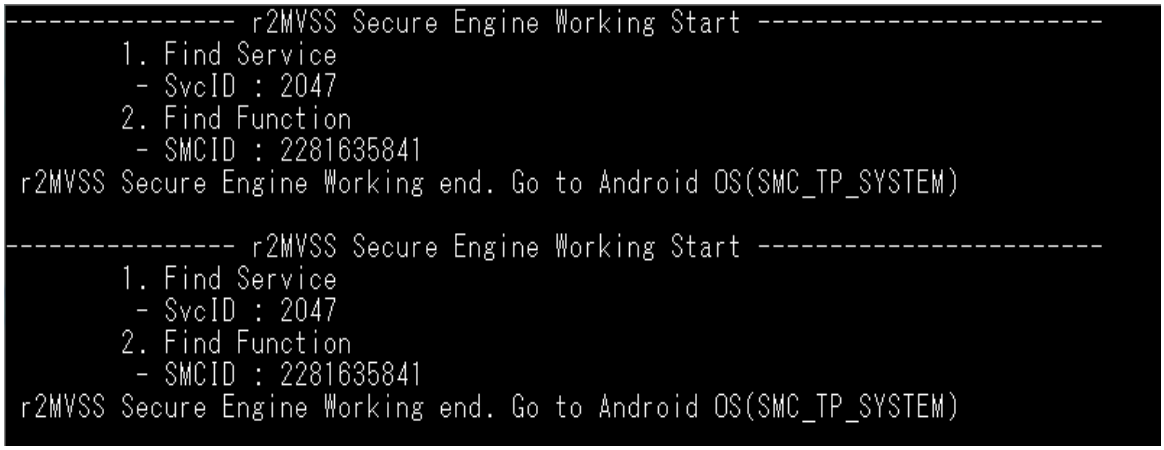

Figure 5. Channel start message in Mobile Virtualization Management Module.

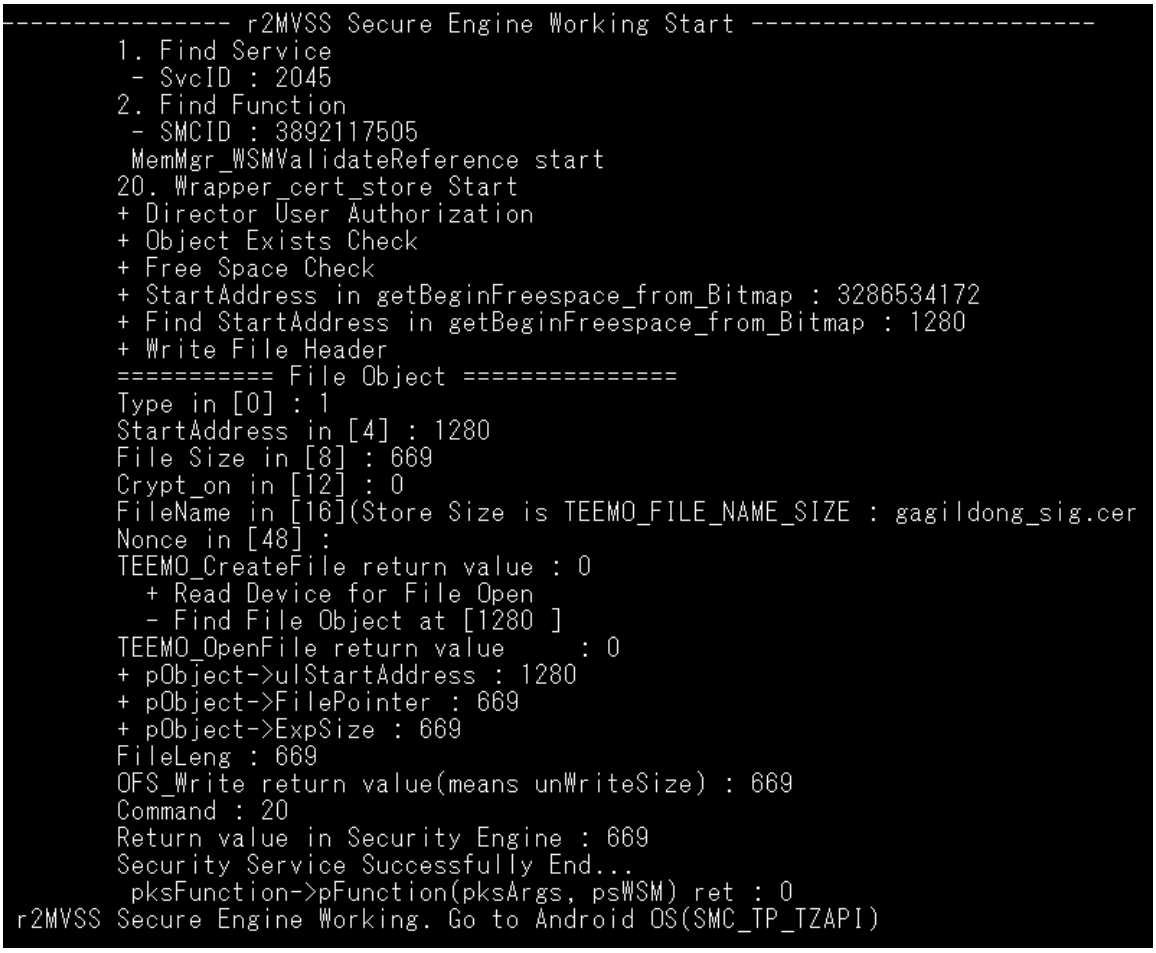

Figure 6. Authentication success message in Secure Authentication Module.

Table 1. Access control authorities according to security policy

Access No	Access authority	OC	OS	HP	SC/SK	GCN	CS	CC	SIGN	Other
22444195	Administrator	O	O	O	O	O	O	O	O	All O
22444194	Manager	O	O	O	X	O	O	O	O	All X
22444193	Common user	O	O	O	X	X	O	O	O	All X
22444192	Guest	O	O	X	X	X	O	O	X	All X
22444191	Director	O	O	O	O	O	O	O	O	INIT_PIN X, GET_KEY_DATA X, and All O
Unknown	Unconfirmed	X	X	X	X	X	X	X	X	All X

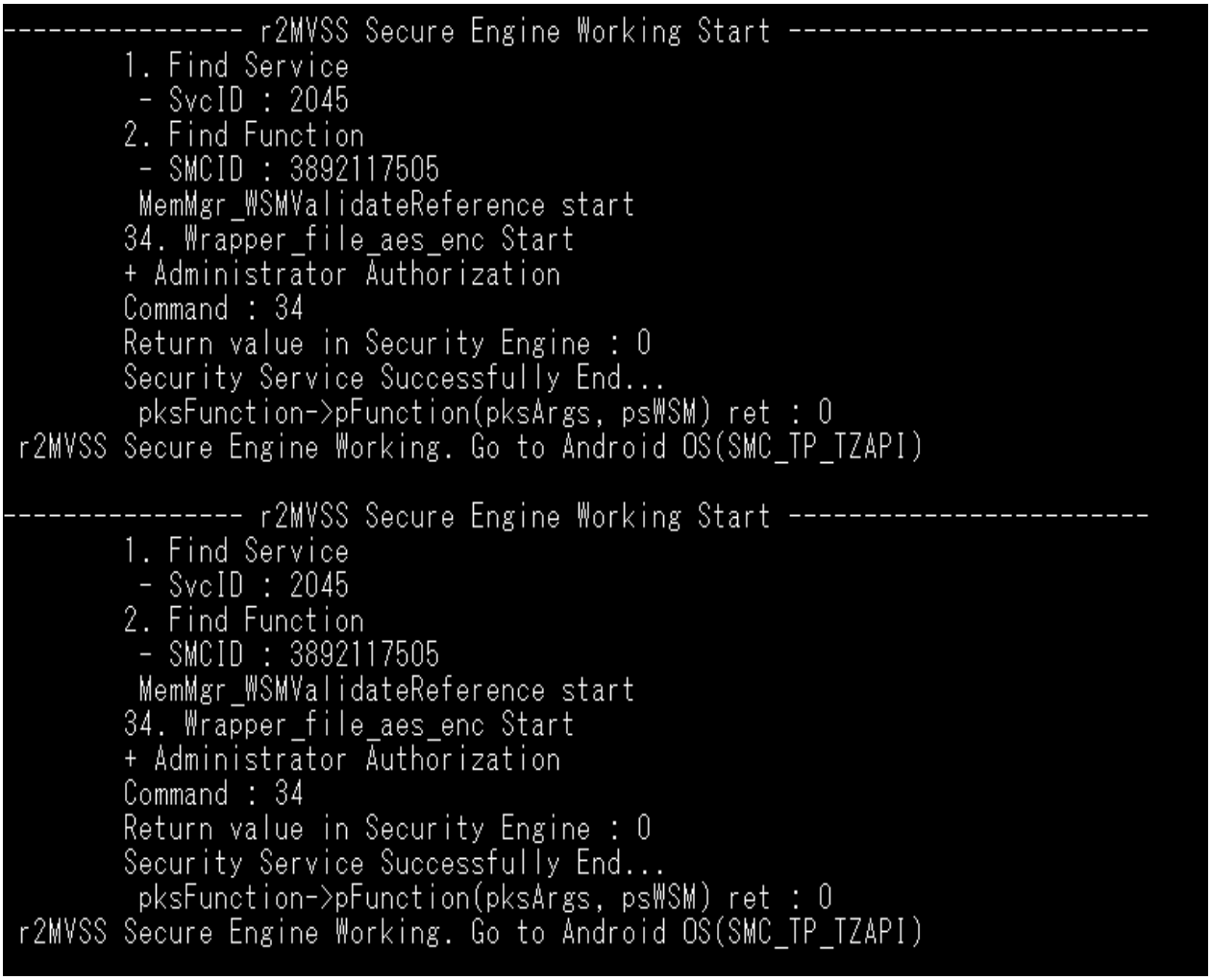

Figure 7. Result message of encryption/decryption with cipher algorithm.

authority execution. Figure 6 shows how authentication is carried out in the secure engine of the secure domain. Table 1 shows items whose access control authority is differentially applied for each user according to the security policy.

In this system, the key function of protecting important data by blocking unauthorized access is managed by cipher algorithms. Various security algorithms are provided in this system including block cipher, public-key cryptography, key exchange, hash function, digital signature, etc. The AES, the ARIA, and the SEED are used for block cipher, and the RSA is used for public-key cryptography. For key exchange, the Diffie-Hellman Algorithm is applied, and the SHA is applied for hash function. Digital signature algorithm is implemented based on the RSA-PSS and the KCDSA to satisfy the conditions of anti-forgery, signer authentication, non-repudiation, inalterability, and non-reusability. Figure 7 shows output message when encryption/decryption functions are performed in the secure engine.

The secure storage function is an essential respect to protect key information from hacking risks in the normal domain by saving information including personal/company data, security required document, key for encryption/decryption in the secure domain where OS is separated. In addition, only authorized application programs can save/take out data in the secure storage in order to restrict discretionary usage of the secure storage.

3.2 Mobile Office Application

Mobile office application installed in the normal domain performs company's tasks which need security factors. It gets authority to access to the secure domain through authentication in process of login and initiation. When executing a function which needs security, it is implemented to proceed the request by calling the secure engine of the secure domain. Detailed functions are electronic approval, in-company bulletin board, personal information storage, email, and company document encryption. These office tasks are safely performed in the mobile device.

4. System Test and Result

4.1 Test Procedure and Condition

In the measurement test on the Type-1-based virtualization security system in the smart devices environment, it has firstly tested whether virtualization through hypervisor of this system is properly implemented, and confirms that security functions run without any malfunctions. Next, we measured the required running times to process services compared with the existing TEEMO Type-2-based virtualization security system. On the security service with a similar function of the two comparative

Table 2. The list of apis and their operating functions used in the measurement test

APIs	API Name	Description
API 01	OS	Open virtualization security session
API 02	OC	Open virtualization security channel
API 03	CC	Close virtualization security channel
API 04	CS	Close virtualization security session
API 05	GC	Get certificate from the secure domain
API 06	RF	Remove files in the secure domain
API 07	GF	Get the number of files in the secure domain
API 08	RP	Register user PIN
API 09	CP	Change user PIN
API 10	FE	Encrypt file
API 11	GSK	Get digital signature with the security algorithm, KCDSA
API 12	GSP	Get digital signature with the security algorithm, PSS

systems, we measured operating times by dividing the situations of general use environment and 50% CPU overload with memory occupation for each API. Last, we checked the average processing time by consecutively calling APIs which have been selected randomly.

The list of APIs and explanation for each function used for the measurement test is shown in Table 2 below:

4.2 Test Result

We tested the operation of security functions with the same environment and condition according to the procedure described above for the two systems, Type-1-based virtualization security system in smart devices environment which is proposed in this paper and existing TEEMO Type-2-based virtualization security, and found the both systems normally provide security service with satisfactory solidity.

The comparison result of processing times of security service APIs in the both systems is shown in the Figure 8. We measured the processing times of general use environment and 50% CPU overload with memory occupation for each system respectively, writing down the average value of five(5) times of measurement. The X-axis indicates each security service API described in the Table 2, and Y-axis means the turnaround time from the moment of performing security process which was requested by the uCOS in the secure domain to the moment of returning the result to the normal domain, after security service APIs are called from the Android OS of the normal domain.

As shown in the Figure 8, security service APIs of the proposed Type-1-based virtualization security system told the processing times from 0.051 to 0.484 seconds, whereas the TEEMO Type-2-based virtualization security system showed from 0.349 to 0.935 seconds of API processing times. It means our proposing system is better at the aspect of processing time of the security service APIs, shortened from 48% to 85%. In the situation of the CPU overload with memory occupation, our system showed the processing speed from 0.097 to 0.659 seconds and comparative system showed from 0.487 to 1.124 seconds. Under the overload situation, for the average increasing ratio of each API processing time, our system indicated 1.70 times, whereas the comparative system showed 1.33 times, which is a normal result as judged with absolute value for increased response time.

Figure 9 shows response times of processing random consecutive calls from security service APIs. It is a measurement graph for 200 times of API calls in a discretionary time point. We can see that overall processing times have increased because request and return for each security service API are consecutively done by the purpose of test measurement, not by a real user application to use the security service. In the selected measurement

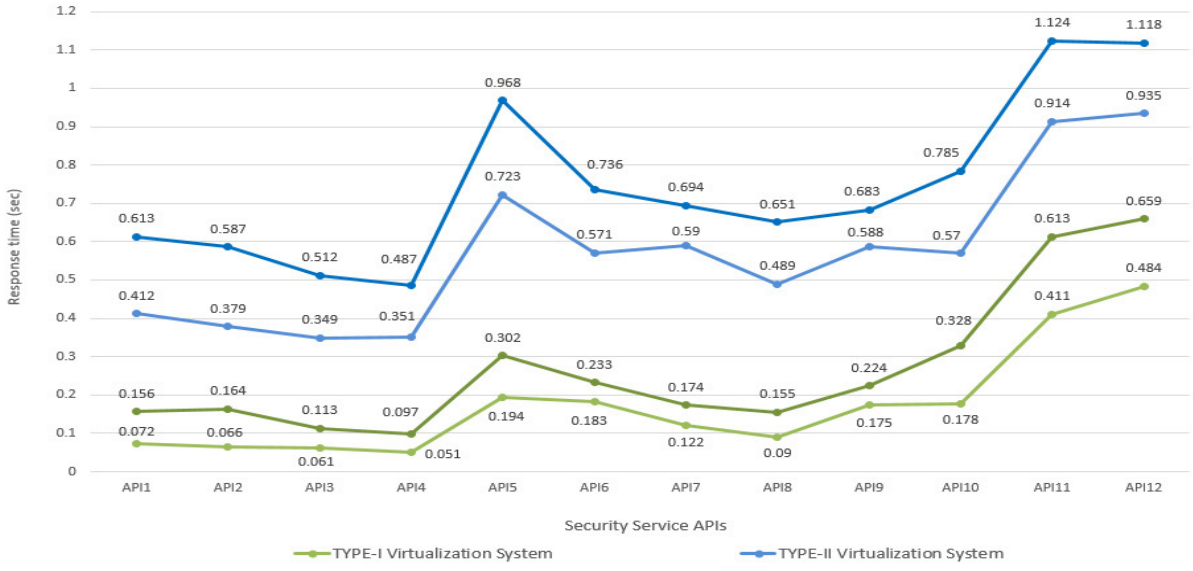

Figure 8. Test result of response time measurement for each security service API.

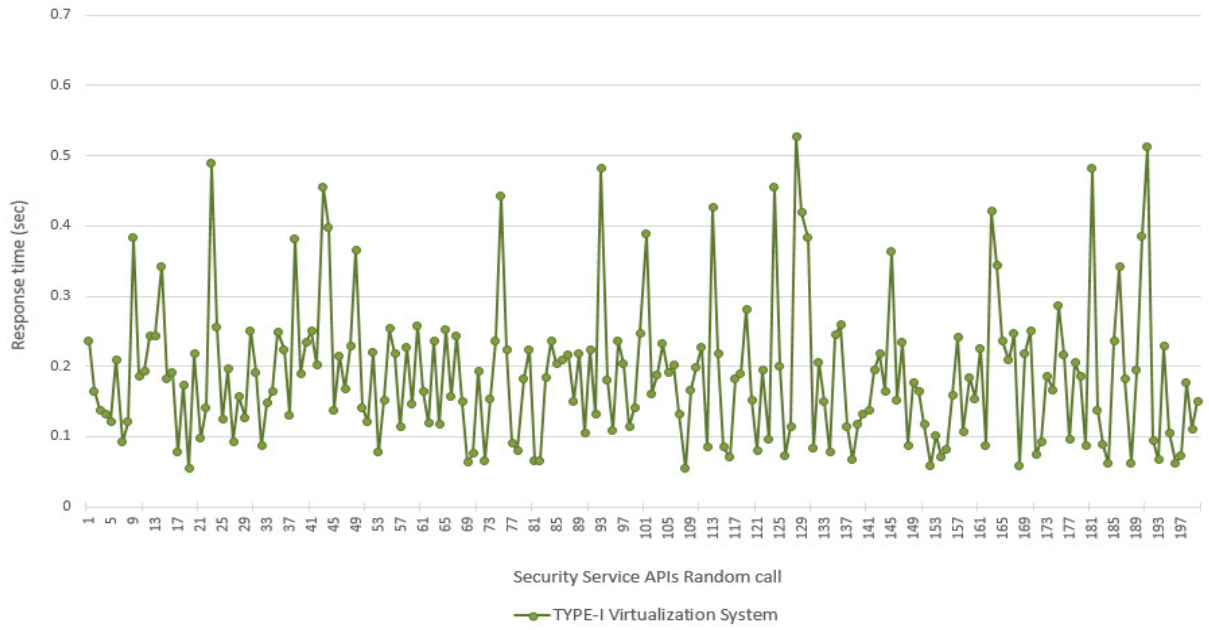

Figure 9. Test result of consecutive calling for security service APIs.

in Figure 9, the time has delayed for 8.62% than a normal situation, the average processing time is 0.189 seconds. In this graph, a striking long response times are file processes of the secure domain or encryption-related APIs. It confirms that our system can provide security service within the time real users can sufficiently accept in the condition of consecutive API calls.

5. Conclusions

As mobile devices are getting more popular, people are increasingly requesting to manage personal information and business work with mobile devices, and interest in mobile security is also increasing. However, mobile security solutions currently used don't have separated security OS environment, contrary to virtualization security system. They don't provide improved security against leakage risks of important data. Therefore, as introduced in this paper, we have researched and implemented Type-1-based virtualization security system and have created a test product which uses a mobile office application.

Security functions such as mobile virtualization management, authentication, encryption/decryption, safe storage have been performed successfully at the test, and we confirmed that it has better processing speed compared with existing Type-2-based virtualization security system. With this research, we expect to help build up necessary security factors and establish secure environment that are needed in the hypervisor virtualization system, noticed as a new security technology recently. In the future, performance and operation ability of mobile devices are expected to improve. It needs to conduct research that categorized applications access to different operating systems with their differentiated security policy according to a required security level by adopting lighter operating systems in virtualization environment.

6. References

1. Young-Ho K, Jeong-Nyeo K. Building Secure Execution Environment for Mobile Platform. Proceedings of 2011 First ACIS/JNU International Conference, Korea. 2011. p. 119 –22.
2. Young-Ho K, Yun-Kyung L, Jeong-Nyeo K. TeeMo: A Generic Trusted Execution Framework for Mobile Devices.

Proceedings of International Conference CNSI 2012, Korea. 2012; 579–583.
3. Seong-Kyeom K, Seung-Jin M. Wireless Intrusion Prevention System (WIPS) growth and build. Proceedings of Korean Institute of Communications and Information Sciences Conference, Korea. 2013. p. 581–2.
4. Jeffrey B, Ryan OH, Arati B, Vinod G, Liviu I. Rootkits on Smart Phones: Attacks, Implications and Opportunities. Proceedings of the Eleventh Workshop on Mobile Computing Systems and Applications, USA. 2010; 49–54.
5. Keunwoo R, Woongryul J, Dongho W. Security Requirements of a Mobile Device Management System. International Journal of Security and its Applications. 2012 Apr; 6(2):353–8.
6. Ronald P, Leendert VD, Reiner S. Virtualization and Hardware-Based Security. IEEE Security and Privacy. 2008 Oct; 6(5):24–31.
7. Paul B, Boris D, Keir F, Steven H, Tim H, Alex H, et al. Xen and the Art of Virtualization. Proceedings of the nineteenth ACM symposium on Operating systems principles, USA. 2003; 164–77.
8. Sung-Min L, Sang-Bum S, Bok-deuk J, Sang-dok M. A Multi-Layer Mandatory Access Control Mechanism for Mobile Devices Based on Virtualization. Proceedings of 2008 5th IEEE Consumer Communications and Networking Conference, USA. 2008. p. 251–6.
9. Tal G, Ben P, Jim C, Mendel R, Dan B. Terra: a virtual machine-based platform for trusted computing. Proceedings of the nineteenth ACM symposium on Operating systems principles, USA. 2003; 193–206.
10. Muthu Pandi K, Somasundaram K. Energy Efficient in Virtual Infrastructure and Green Cloud Computing: A Review. Indian Journal of Science and Technology. 2016 Mar; 9(11). DOI: 10.17485/ijst/2016/v9i11/89399.
11. Durairaj M, Manimaran A. A Study on Security Issues in Cloud based E-Learning. Indian Journal of Science and Technology. 2015 Apr; 8(8):757–65.
12. Jae-Deok L, Jeong-Nyeo K. A Study on the Trusted App.-based Access Control to the Isolated Trusted Execution Environment in Mobile Device. Proceedings of the Korean Institute of Communications and Information Sciences Conference, Korea. 2014. p. 364–5.
13. Ahmed MA, Peng N, Jitesh S, Quan C, Rohan B, Guruprasad G, et al. Hypervision Across Worlds: Real-time Kernel Protection from the ARM TrustZone Secure World. Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, USA. 2014. p. 90–102.
14. Hwi-Min C, Chang-Bok J, Joo-Man K. Efficient security method using mobile virtualization technology and trust-zone of ARM. Journal of Digital Convergence. 2014 Oct; 12(10):299–308.
15. Nuno S, Himanshu R, Stefan S, Alec W. Using ARM trustzone to build a trusted language runtime for mobile applications. Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems, USA. 2014. p. 67–80.
16. Joo-Young H, Sang-Bum S, Sung-Kwan H, Chan-Ju P, Jae-Min R, Seong-Yeol P, et al. Xen on ARM: System Virtualization using Xen Hypervisor for ARM-Based Secure Mobile Phones. Proceedings of 2008 5th IEEE Consumer Communications and Networking Conference, USA. 2008. p. 257–61.
17. Seehwan Y, Chuck Y. Real-Time Scheduling for Xen-ARM Virtual Machines. IEEE Transactions on Mobile Computing. 2014 Aug; 13(8):1857–67.