Security Analysis of Samsung Knox

Munkhzorig Dorjmyagmar, MinChang Kim, Hyoungshick Kim
Department of Computer Science and Engineering, Sungkyunkwan University, Korea
Email: {mnkhzrg, mckim, hyoung}@skku.edu

***Abstract*—A Trusted Execution Environment (TEE) has become popular in the mobile industry. Hardware-based security will be employed by default for every mobile device within a few years. In this paper, we explore several potential security issues of the `Samsung Knox` platform that is one of the advanced hardware based mobile security platforms for Android devices. We describe several attack scenarios to show how the Knox platform can be compromised. We particularly performed experiments for Man in the Middle Attacks with an untrusted certificate. To mitigate such security risks, we also recommend several countermeasures based on fundamental security principles. For example, security-sensitive resources in Knox should be strictly isolated from processes in an insecure operating system.**

***Keywords*—TrustZone, Samsung Knox, Trusted Computing.**

I. Introduction

As smartphones have become indispensable in people's everyday lives, smartphones are also increasingly becoming the attractive targets of malware. To protect a user's sensitive data on his or her smartphone from malware, a Trusted Execution Environment (TEE) is typically required [1] because the processes and resources in TEE are securely isolated from insecure processes [2]. In general, TEE is implemented based on a trustworthy hardware component. Recently, the most popularly used solutions are Trusted Computing (TC) [3], Intel SGX [4] and ARM TrustZone. In particular, many enterprises such as Microsoft, Google and IBM have become interested in those solutions to successfully implement a *Bring Your Own Device* (BYOD) environment [5]. Various security threats against BYOD can be mitigated by running the enterprise software inside a *secure container* on employees' smartphones.

Recently, Samsung introduced a mobile platform for secure containers called *Knox* to mitigate the threats related to BYOD. Knox can be used to provide a trusted execution environment for BYOD environments using the ARM TrustZone technology.

In this paper, we discuss several security threats in Samsung Knox. To show the feasibility of those threats in real world situations, we particularly implemented some sophisticated attacks called *Man-In-the Middle* (MITM) attack.

The rest of the paper is organized as follows. Section II briefly reviews the representative technologies (TC, ARM TrustZone, Intel SGX and Samsung Knox) for trustworthy computing. Section III introduces security vulnerabilities and possible attack scenarios in Samsung Knox. Section IV demonstrates our attack implementation for MITM attacks. Finally, we conclude the paper along with future work in Section V.

II. Trustworthy computing technologies

A. Trusted Computing

The idea of TC was introduced to provide a TEE in both mobile and PC domains [6]. With TC, the devices will consistently behave in the expected ways, and those behaviors will be enforced by computer hardware and software. TEE offers an execution space that provides a higher level of security in a rich mobile operating system.

B. ARM TrustZone

TEE [1] is an isolated environment that runs in parallel with the mobile OS, providing security for the rich environment. Each of the physical processor cores provides two virtual cores; non-secure (normal world) and secure (secure world), and a mechanism for robustly context switch between them, known as Secure Monitor Call (SMC). Non-secure virtual processor only accesses non-secure system resources, but secure virtual processors can see all resources. These two virtual processors execute in a time-sliced fashion [7].

C. Intel SGX

An enclave of SGX [4] is equivalent to a Trustlet of the TrustZone, yet the only key difference is the way how each of them run. An Enclave is run from within a regular process runtime context in dedicated secure memory without changing the processor mode. SGX is a TEE for Intel processors supporting the execution of trusted code in secure containers called enclaves by employing dedicated hardware and processor instructions. An enclave is executed from within a regular process runtime context in dedicated secure memory without changing the processor mode [4]. We briefly explain the functionality of each component in the Knox platform as follows (see Figure 1).

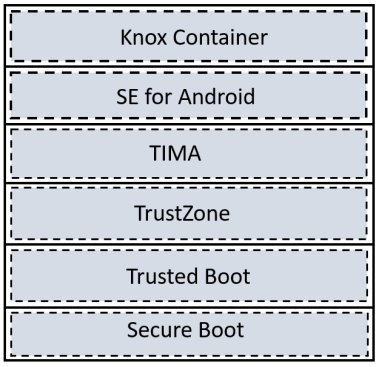

Fig. 1. Knox multilayer platform.

D. Samsung Knox

Samsung Knox is a multi-layered and defense-grade mobile security platform built into Samsung's flagship mobile devices. The overview of the Knox architecture is shown in Figure 2.

1) Secure boot sequence: The first security level of hardware is the secure boot. It prevents unauthorized boot loaders and detects the device OS whether it is cryptographically signed by a key verification hardware. The boot starts from the primary bootloader stored in ROM. The secondary bootloader verifies and loads the secure world OS. The secure world in turns, runs Trustzone-based Integrity Measurement (TIMA) which verifies the integrity of the kernel. If any of the boot components have been tampered, the "Knox Warranty Bit" will be turned on and Knox will be terminated immediately.

2) SE for Android: To prevent serious damages from hacking and malware [8], security enhancements for android isolates applications and data into two different domains based on confidentiality and integrity requirements. So it can prevent a security compromise in one domain from propagating to other domains. To this end, multiple proprietary secure world OS implementations for SE for Android, such as QualComm Secure Execution Environment (QSEE) and Mobicomer on the Samsung devices.

3) TrustZone-based integrity measurement architecture: Once mobile is booted, TrustZone-based integrity measurement architecture (TIMA) runs in the secure world of the TrustZone. It provides continuous integrity monitoring of the OS by constantly checking the Android Kernel. Precisely, it is a set of Trustlets, running within the TrustZone that provides the basis of a secure boot, ensures the system's integrity at runtime and provides security critical services. TIMA has two main components as below:

- Periodic Kernel Measurement (PKM). It is a TIMA component that periodically performs validations on the kernel code and data.
- Real-time Kernel Protection (RKP). It is the core of KNOXs runtime security, guarding against kernel corruption at run-time.

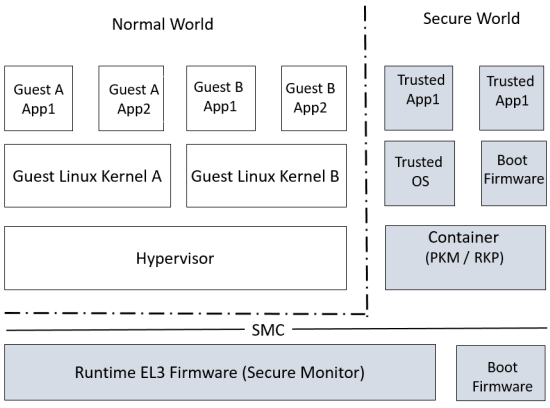

Fig. 2. Knox architecture.

4) Attestation: Attestation is an enterprise crucial feature for situations where an external application may want to validate the device's condition before allowing it to download sensitive data. The attestation is performed within the TrustZone and produces a token that can be cryptographically linked to the device indicating whether it has been compromised. The Device Root Key (DRK) is a unique asymmetric key that is signed by Samsung through an X.509 certificate.

III. Attack scenarios

In this section we describe five known vulnerabilities and attack scenarios. Further, we determine some of possible attack scenarios against Knox 1.0. We divide the attacks into two categories, root privilege required and not required. Each attack scenario is explained as below:

A. Clipboard data exposure

CVE-2016-3996 [9] is a vulnerability that allows an attacker to intercept the contents of the Knox clipboard. ClipboardEX is the proprietary services related to Knox. Knox application accesses clipboard via ClipboadExManager class. The clipboardEx Manager is not available through the Android SDK for the developers. In order for an attacker to get access to the Knox clipboard, all that is needed is for the service to have "mContainerID != 0".

B. Man in the Middle Attack

CVE-2016-1920 [9], a vulnerability which allows a user application running outside Knox to perform a Man-in-the-Middle (MITM) attack against Knox SSL/TLS traffic. Android manages X.509 certificates through a certificate storage. This storage is used when the validity of an SSL/TLS certificate was checked for network hosts (e.g., a website).

Android supports installing third party certificates. Then side-effect of the service sharing in the Knox 1.0 design is that the same certificate store applies to both Android and Knox applications.

C. eCryptFS key exposure

By revealing the vulnerability CVE-2016-1919 [9] an attacker can decrypt Knox encrypted data without knowing the users password. As specified by the Knox 1.0 Whitepaper, the key used for the encryption is derived by "key-derivation algorithms, such as Password-Based Key Derivation Function 2 (PBKDF2)". PBKDF2 generates the Master Key, but it is rendered meaningless due to the poor generation of the eCryptFS key.

D. Android Debug Bridge (ADB) attack

This allows an attacker having a foothold on the users computer to attack Knox via USB if it is unlocked. Using ADB, an attacker can launch the Knox secure world browser with a target URL leading to a website under the attackers control. An additional security hole is the ability to send broadcast that will be received by Knox applications as well.

E. Screen capture

In Periodic Detection we have shown that if a malicious root attacker can get through Knox, he can extract some of sensitive information and corrupt the system. Remote code execution based vulnerability, the Linux kernel API based vulnerability, and Knox out [10] are also recently resolved vulnerabilities by Samsung.

IV. Experiments

A. Procedure of VPN MITM attack

VPN MITM attack allows attackers to disable the Knox security by taking an advantage of the vulnerability where secure world and normal world shares a common certificate storage. In general, Knox will alarm when the attacker executes any malicious application. The basic idea of VPN MITM attack is shown in Figure 3.

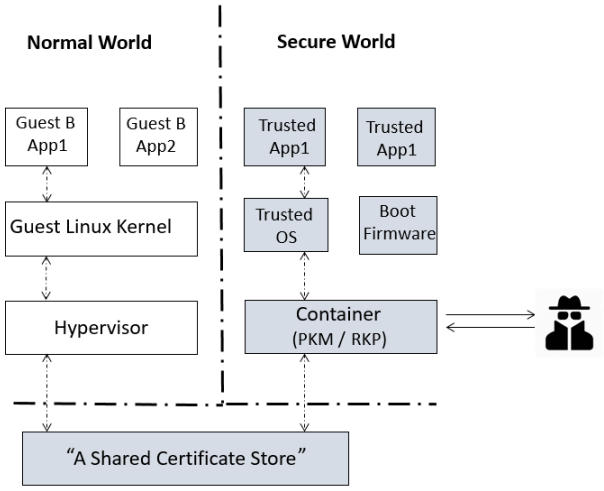

Fig. 3. VPN MITM attack.

We explain how the VPN MITM attack can be performed as follows:

- Android manages X.509 certificates through a certificate store. This store is used when validating SSL/TLS certificates using the chain-of-trust (Bottom to Up Validation method) scheme.
- Android allows installing third party certificates with the help of user interaction.
- In the Knox 1.0 design, one of the security hole is service sharing, which the same certificate store applies to both Normal world and Knox applications. That means, once the server certification is validated in SSL/TLS connection, Knox application relying on chain-of-trust verification will trust any third party certificates installed by user through the normal world.
- The VPN feature in Android allows an application to register as a VPN provider and route all traffic through it. This involves asking a permission upon installation and VPN connection startup.

B. VPN MITM attack via SMS application

To experiment with VPN MITM attacks, a Samsung smartphone with the KNOX 1.0 version. Galaxy S3 SHV-E210S was used for our experiments.

To deploy VPN MITM attack in a more realistic setting, we can use SMS messages that were shared between normal world and secure world. We observed that SMS messages are shared between normal world and secure world by default. When a SMS message is delivered to a victim's smartphone, SMS applications in both worlds can access the received SMS message simultaneously. Therefore, SMS messages can be used for triggering a VPN MITM attack.

Our attack scenario is as follows:

- Install a malicious application with VPN-related permissions.
- Install the third party certificate for the malicious application in normal world.
- Send the link for a malicious website via SMS. When a victim click the link even in a browser application in secure world, the malicious website is successfully loaded.

This attack scenario is also shown in Figure 4. The notification "red flag" can be easily mitigated by the attacker. By using Knox icon and a benign name for the VPN connection such as "Knox Connectivity" a not-tech user will take the notification as a legitimate part of Knox and continue using the device normally. By default the user can receive SMS message on both normal world and secure world simultaneously. Taking advantage of the shared resources, we can easily deploy a VPN MITM attack.

C. Countermeasures

Advancing more integration with the following countermeasures can mitigate the above mentioned vulnerabilities. These countermeasures are well known mechanisms for reducing security risks as these are the principal software engineering concepts.

1) Managed code protection: Given that Knox utilizes managed (Java) code as a part of its security infrastructure (system server) and allows to run Java-based applications, it must adequately protect the managed-code layer. Otherwise, without proper protection the secure container becomes highly vulnerable to the code-injection attacks.

2) Reuse of components: One of the well-designed implementation that Knox employs is the reuse component. Running Knox side-by-side with the regular Android applications, separating the processes using SELinux is a reasonable design choice given that it reuses an existing security feature of the operating system [11].

3) Application validation: Although Application wrapping exists in Knox 1.0 only due to a technical limitation it is an excellent security feature. As already proven by iOS, compared to Android, a thorough process of application review by a trusted party greatly diminishes the risk of a malware making it to the application store [11].

4) Limiting the attack surface: It is difficult to design a secure container against all possible attacks. However, it seems helpful to disallow users to perform potentially dangerous activities for narrowing down the secure container's attack surface. For example, to perform an ADB attack, an attacker needs to use the communication method via ADB. Therefore, our first line of defense could be disable ADB debugging when the Knox mode is active.

5) Avoiding resource sharing: Sharing resources between the secure container and the insecure environment is a recipe for a disaster. It has been observed through the sharing of the connectivity service and the certificate store in the VPN

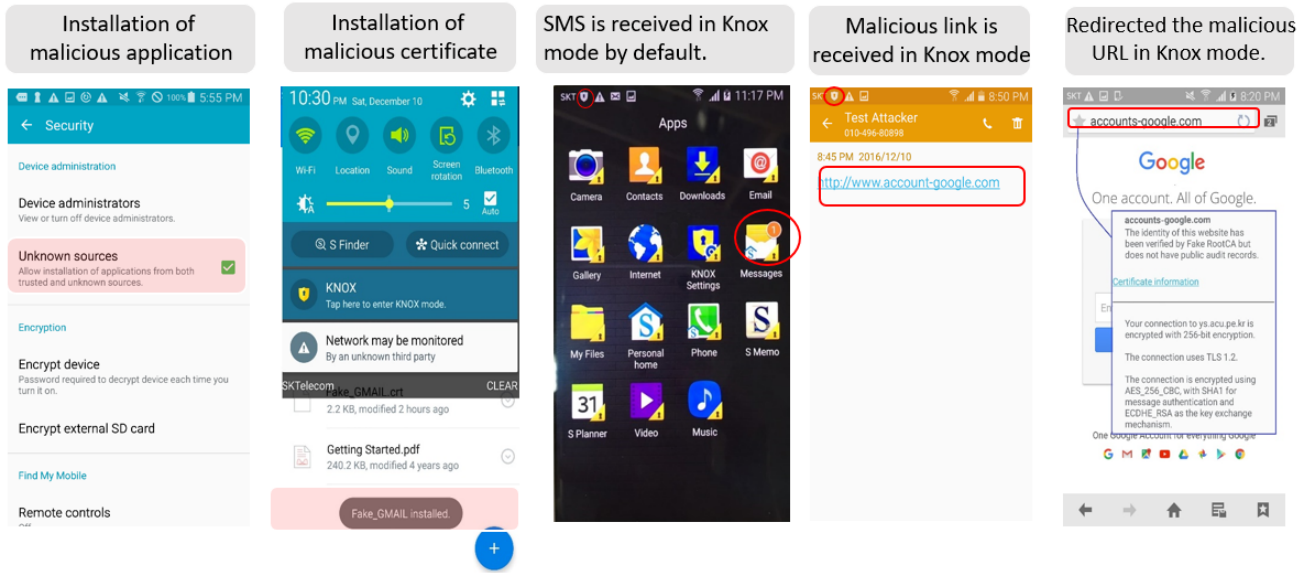

Fig. 4. VPN MITM attack via an SMS message.

attack. In particular, any resource can be either monitored or modified. Among the above mentioned attack scenarios "Root privileges not required attacks" are more critical issues than other type of attack scenarios.

V. Conclusion

In this paper, we analyzed the container based security solution called Knox for Samsung's flagship devices.

To show the feasibility of our observations, we particularly implemented a prototype attack for VPN MITM attack. Probably, however, those attacks could be mitigated by pursuing fundamental security principles. We discussed several mitigation techniques to fix those vulnerabilities. As an extension to this paper, we plan to implement those mitigation techniques and evaluate their performance against the attacks.

References

[1] R. van Rijswijk-Deij and E. Poll, "Using Trusted Execution Environments in Two-factor Authentication: comparing approaches," in *Open Identity Summit*, 2013.
[2] "Android security white paper," Google, Tech. Rep., 2016.
[3] N. Asokan, J. E. Ekberg, K. Kostiainen, A. Rajan, C. Rozas, A. R. Sadeghi, S. Schulz, and C. Wachsmann, "Mobile Trusted Computing," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1189–1206, 2014.
[4] V. Costan and S. Devadas, "Intel SGX explained," *IACR Cryptology ePrint Archive*, 2016. [Online]. Available: http://eprint.iacr.org/2016/086
[5] A. Armando, G. Costa, and A. Merlo, "Bring Your Own Device, Securely," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, 2013.
[6] J. M. McCune, B. J. Parno, A. Perrig, M. K. Reiter, and H. Isozaki, "Flicker: An Execution Infrastructure for Tcb Minimization," *SIGOPS Operating Systems Review*, vol. 42, no. 4, pp. 315–328, 2008.
[7] *ARM Security Technology - Building a Secure System using TrustZone Technology*, ARM Limited, 2009.
[8] M. Xu, C. Song, Y. Ji, M.-W. Shih, K. Lu, C. Zheng, R. Duan, Y. Jang, B. Lee, C. Qian, S. Lee, and T. Kim, "Toward engineering a secure android ecosystem: A survey of existing techniques," *ACM Computing Surveys*, vol. 49, no. 2, pp. 38:1–38:47, 2016.
[9] U. Kanonov and A. Wool, "Secure Containers in Android: The Samsung KNOX Case Study," in *Proceedings of the 6th Workshop on Security and Privacy in Smartphones and Mobile Devices*, 2016.
[10] L. Aronsky, "KNOXout – Bypassing Samsung KNOX," VIRAL Security Group, Tech. Rep., 2016.
[11] R. Pressman, *Software Engineering: A Practitioner's Approach*, 7th ed. McGraw-Hill, Inc., 2010.

Munkhzorig Dorjmyagmar was born in Ulaanbaatar, Mongolia, in 1982. He received his Bachelor degree in Information Technology from Uttar Pradesh Technical University, Lucknow, India, in 2007. He is currently pursuing his M.S. degree in Electrical and Computer Engineering at Sungkyunkwan University, Suwon, Korea. He has been working in the Technology and Research Institute of Mongolia since 2008. His research interests are security engineering, mobile platform security and social network analysis.

MinChang Kim received his M.S. degree in business administration from Sungkyunkwan University, Seoul, Korea, in 2009. He is currently pursuing a Ph.D. degree in Electrical and Computer Engineering at Sungkwyunkwan University, Suwon, Korea. Since 2006, he has been working in the Software Content Research Laboratory of Electronics and Telecommunications Research Institute. His research interests are security engineering, mobile security, IoT Security, and artificial neural network.

Hyoungshick Kim Hyoungshick Kim is an assistant professor in the Department of Computer Science and Engineering, College of Information and Communication Engineering, Sungkyunkwan University. He received a BS degree from the department of Information Engineering at Sungkyunkwan University, a MS degree from the Department of Computer Science at KAIST and a Ph.D. degree from the Computer Laboratory at University of Cambridge in 1999, 2001 and 2012, respectively. After completing his PhD, he worked as a post-doctoral fellow in the Department of Electrical and Computer Engineering at the University of British Columbia. He previously worked for Samsung Electronics as a senior engineer from 2004 to 2008. He also served as a member of DLNA and Coral standardization for DRM interoperability in home networks. His current research interest is focused on software security and usable security.