Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Improving mobile device security with operating system-level virtualization

Sascha Wessel[*], *Manuel Huber*, *Frederic Stumpf*, *Claudia Eckert*

Fraunhofer Institute AISEC, Germany

ARTICLE INFO

Article history:
Received 13 November 2014
Received in revised form
21 January 2015
Accepted 12 February 2015
Available online xxx

Keywords:
Operating system-level virtualization
Mobile device security
Integrity protection
Smartphones
Android

ABSTRACT

In this paper, we propose a lightweight mechanism to isolate one or more Android userland instances from a trustworthy and secure entity. This entity controls and manages the Android instances and provides an interface for remote administration and management of the device and its software. We provide an administrative solution for dynamically modifying, removing or adding multiple Android instances remotely and locally. Furthermore, we present a secure device provisioning and enrollment solution for our system. Our approach includes several security extensions for secure network access, integrity protection of data on storage devices, and secure access to the touchscreen of mobile devices. Our implementation requires only minimal modification to the software stack of a typical Android-based smartphone, which allows easy porting to other devices when compared to other virtualization techniques. Practical tests show the feasibility of our approach regarding runtime overhead and battery lifetime impact.

1. Introduction

Smartphones are already an omnipresent part of our everyday lives. They are used for various tasks with different security requirements like web browsing, banking, or business use cases. This results in an increased demand for isolated environments with different security levels for different tasks on a single device. Payment service providers want a secure environment to protect their applications for financial transactions. Companies want a corporate environment isolated from the private environment of a user and the possibility to manage the devices remotely. This especially includes the enforcement of various security policies, which cannot be enforced with a stock Android-based smartphone today, e.g., whitelisting and/or blacklisting of applications and versions of applications in case of known vulnerabilities.

In this contribution, we propose a lightweight isolation mechanism for Android based on operating system-level virtualization and access control policies to separate one or more Android userland instances from a trustworthy and secure environment. Furthermore, we propose several security extensions based on this environment to control and manage the Android instances and their input and output data. This includes secure network communication, integrity protection of data on storage devices, and secure access to the touchscreen, e.g., for password entry dialogs. Another important part of our security concept is the integration of a secure element (SE), embedded into a microSD card, connected via near field communication (NFC) or bluetooth low energy (BTLE), to store secret keys and private data physically separated from the application processor of the smartphone. This is to ensure its protection even in case of hardware-based attacks. Furthermore, our concept aims at straightforward

* *Corresponding author*. Tel.: +49 89 3229986 155; fax: +49 89 3229986 299.
E-mail address: sascha.wessel@aisec.fraunhofer.de (S. Wessel).

http://dx.doi.org/10.1016/j.cose.2015.02.005
0167-4048/© 2015 Elsevier Ltd. All rights reserved.

remote manageability for integration in IT infrastructures with many users and devices. This includes easy snapshot and recovery functionalities and, moreover, security updates independent of the smartphone manufacturer. A public key infrastructure (PKI) with different certificates for devices, user identities, software signatures and for a backend system is used for authentication, signing and/or data encryption. In order to improve the security of the product from the beginning of its lifecycle, we also present a secure device provisioning and enrollment solution. The evaluation of our prototype implementation shows that our modifications to the Android software stack introduce only a negligible performance overhead and reduce the battery lifetime by only 7.5 percent in the worst case.

This paper is organized as follows. In Section 2, we introduce our attacker model and in Section 3 an overview of virtualization techniques for Android is given, followed by a discussion of related work in Section 4. Section 5 introduces the basic concept of operating system-level virtualization for Android and our additional security mechanisms are described in Section 6. General implementation aspects are presented in Section 7 and an overview of our system architecture is presented in Section 8. A PKI with all involved certificates is proposed in Section 9, whereas secure device provisioning and enrollment is focused on in Section 10. Our prototype is described in Section 11. Finally, we evaluate our results in Section 12 and conclude in Section 13.

2. Attacker model

In our attacker model, we assume an attacker using common attack vectors on Android-based smartphones. This especially includes eavesdropping and modification of remote communication, installation of malicious applications on the device, and exploiting (known) vulnerabilities to gain access to higher privilege levels, typically root access. The attacker is thus capable of threatening the device by all means of communication. Breaking common cryptography for signing, authentication and encryption is not possible in our model.

Besides remote attackers, we also consider local attackers with physical access to the mobile devices. However, it is not possible to protect the data and software running on the application processor against attacks using JTAG or similar mechanisms without modifications to the smartphone hardware, so such attacks are out of scope for this paper. Furthermore, we assume that the attacker is not capable to extract private keys from SEs.

3. Virtualization techniques for Android

Isolation mechanisms or rather virtualization techniques for Android can be classified in three groups, namely user-level isolation, operating system-level virtualization and system virtualization. Fig. 1 shows these three basic concepts for an example system with two isolated groups of applications and an additional control and management entity. By default, an Android system consists of an application layer, a middleware layer, and the kernel layer on top of the hardware. As shown in the figure, the main difference between the three concepts is which layers are shared by the isolated environments.

User-level isolation is for example achieved in stock Android devices where the applications run as different users. Thus, applications are isolated from each other. System virtualization requires a hypervisor, which provides the possibility to operate multiple isolated operating systems as a whole. Our concept is based on the architecture shown in the middle and is described in Section 5. Operating system-level virtualization aims to virtualize operating system (OS) resources, e.g. via namespaces. Therefore, our technique allows to run different Linux-based guest operating systems, such as Android or Firefox OS in different isolated environments on top of the same kernel. We thus do not focus on isolation between applications or application groups with mandatory access control (MAC) mechanisms as many other papers in this context do and what we would categorize as user level isolation. Instead, it is possible to run all three isolation mechanisms stacked on the same system. User-level isolation as achieved on stock Android devices does not provide the desired level of security, as a common attack on a vulnerability results in full control over the system. To raise the security level, more sophisticated approaches in user-level isolation were therefore developed (Bugiel et al., 2013), see Section 4. On the other side, system level isolation in contrast to stock Android devices, provides the desired level, as a breach in one operating system (OS) has no effect on the runtime of the other OS. However, system virtualization requires major changes to the system and is often associated with lacks in performance, as discussed in Section 4. Operating system-level virtualization combines the advantage of being a lightweight isolation mechanism and ensuring the desired level of security, as a common exploit does not break through container boundaries.

4. Related work

The default Android security architecture takes usage of different user identiers (UIDs) per application group to implement a sandboxing mechanism. The communication between applications and core Android components is restricted based on permissions, which are requested during the installation of applications. It was shown that these mechanisms do not meet all security requirements (Barrera et al., 2010; Davi et al., 2011), which led to a number of extensions to the Android architecture (Bugiel et al., 2011; Shabtai et al., 2010; Ongtang et al., 2010). In contrast to our approach, these user-level isolation mechanisms usually require massive modifications to Android userspace components and introduce more complexity to the overall system. Furthermore, these solutions are usually limited to one type of operating system (e.g. Android or Firefox OS). More recent user-level isolation approaches like FlaskDroid (Bugiel et al., 2013) or MOSES (Russello et al., 2012) seek to achieve a high level of security, meanwhile remaining lightweight and flexible. FlaskDroid proposes efforts in multiple-layer policies, providing flexible and application-specific access control semantics. This secures an Android instance due to the support