FIGI

FINANCIAL INCLUSION
GLOBAL INITIATIVE

SECURITY, INFRASTRUCTURE AND TRUST WORKING GROUP

Security analysis of the KaiOS feature phone platform for DFS applications

REPORT OF THE SECURITY WORKSTREAM

Security, Infrastructure and Trust Working Group

Security analysis of the KaiOS feature phone platform for DFS applications

DISCLAIMER

The Financial Inclusion Global Initiative (FIGI) is a three-year program implemented in partnership by the World Bank Group (WBG), the Committee on Payments and Market Infrastructures (CPMI), and the International Telecommunication Union (ITU), funded by the Bill & Melinda Gates Foundation (BMGF) to facilitate the implementation of country-led reforms to attain national financial inclusion targets, and ultimately the global 'Universal Financial Access 2020' goal. FIGI funds initiatives in three countries-China, Egypt and Mexico; supports working groups to address three distinct challenges for reaching universal financial access:

(1) the Electronic Payment Acceptance Working Group (led by the WBG),

(2) The Digital ID for Financial Services Working Group (led by the WBG), and

(3) The Security, Infrastructure and Trust Working Group (led by the ITU).

FIGI hosts three annual symposia to assemble national authorities, the private sector, and other relevant stakeholders to share emerging insights from the Working Groups and country level implementation.

This report is a product of the FIGI Security, Infrastructure and Trust Working Group, led by the International Telecommunication Union. The findings, interpretations, and conclusions expressed in this work do not necessarily reflect the views of the Financial Inclusion Global Initiative partners including the Committee on Payments and Market Infrastructures, the Bill & Melinda Gates Foundation, the International Telecommunication Union, or the World Bank (including its Board of Executive Directors or the governments they represent). The mention of specific companies, or of certain manufacturers' products does not imply that they are endorsed nor recommended by ITU in preference to others of a similar nature that are not mentioned. Errors and omissions excepted, the names of proprietary products are distinguished by initial capital letters. The FIGI partners do not guarantee the accuracy of the data included in this work. The boundaries, colours, denominations, and other information shown on any map in this work do not imply any judgment on the part of the FIGI partners concerning the legal status of any country, territory, city or area or of its authorities or the endorsement or acceptance of such boundaries.

About this Report

This report was written by Sébastien Mathieu and Philippe Oechslin. The report was reviewed by the FIGI Security Infrastructure and Trust Working Group. The authors would like to thank Vijay Mauree and Arnold Kibuuka, TSB, ITU who provided comments and feedback to the authors on the report.

If you would like to provide any additional information, please contact Vijay Mauree at tsbfigisit@itu.int

Contents

Security analysis of the KaiOS feature phone platform for DFS applications

1 INTRODUCTION

1.1 Context

The goal of this report is to give an overview of the security issues related to DFS when used in simple feature phones running the KaiOS operating system, as opposed to smart phones. Security of DFS applications is one subject of the Financial Inclusion Global Initiative (FIGI) Infrastructure and Trust Working Group.

1.2 Feature phones

Mobile phones can be divided into three categories

a) Basic phones
b) Smart phones
c) Feature phones

1.2.1 Basic phones

At the lower end are the basic phones that have little functionality other that calling and SMS. Although they may have some pre-installed applications, it is not possible to install third party applications on basic phones. DFS could be implemented using SMS or USSD (dialing of specific codes like *#123#).

Beafone C65

1.2.2 Smart phones

Smart phones are phones with an advanced set of hardware and software capabilities. Typical features are a touch screen, fingerprint readers, a hardware security module for storing secrets, fast and diverse data communication technology and a choice of millions of third-party applications.

Iphone

1.2.3 Feature phones

Feature phones are the intermediate ground between basic and smart phones. They are built to be affordable but still allow the installation of third-party applications. They lack hardware features like a touch screen, a fingerprint reader or a hardware security module. Hundreds third party applications can be installed from application stores. Most importantly, feature phones are the cheapest way to access the Internet and are seen as an important tool in bridging the digital divide.
Either feature phones can be used to access DFS by installing the corresponding app on the phone or accessing a web based DFS.

JiO phone running Whatsapp on KaiOS[1].

2 SECURITY RECOMMENDATIONS FOR DFS

To assess the security of KaiOS for DFS, a set of security criteria required for DFS need to be defined. The recommendations on DFS put forth in the report on the Security Aspects of Digital Financial Services[2] of the ITU-T Focus Group Digital Financial Services can be used as security criteria to assess the security of KaiOS operating system. The report makes twenty recommendations. Below are the nine recommendations specifically dealing with the operating system and the software architecture of a phone.

R 1. Consider the use of strong authentication mechanisms to demonstrate ownership of the device.
R 2. Make use of hardware and software mechanisms within mobile devices, such as secure elements and TEEs, which can ensure device integrity, and promote the use of devices equipped with security features for use in DFS.
R 3. Whether an application is designed for deployment on the handset or secure element, it should be designed and implemented in accordance with best practices, including encrypted and authenticated communication and secure coding practices to harden the app.
R 4. Apps should be subjected to external security review and penetration testing, and any recommendations acted upon.
R 5. Apps should securely manage username and password information so that adversaries cannot easily forge credentials, and should use strong authentication mechanisms to protect against unauthorized access.
R 6. Regular security updates are critical to ensure that mobile operating systems running on user devices operate using the latest security patches.
R 7. Ensure that security libraries offered by the operating system are correctly designed and implemented and that the cipher suites they support are sufficiently strong.
R 8. The handset operating system should be configured in a way to reduce the size of the trusted computing base.
R 9. Consider transitioning away from mobile applications that leverage SMS and USSD in favor of solutions that use strong public key cryptography and end-to-end security.

3 KAIOS AND ITS SECURITY MODEL

3.1 KaiOS

KaiOS is an operating system for feature phones that is developed by KaiOS Technologies. It builds on the Firefox Operating System (OS) that was developed by the Mozilla foundation.
The basic idea of KaiOS is that the phone only runs a web browser and that all applications on the phone are written like web applications in HTML5, JavaScript and CSS. Thus, the browser almost acts like an operating system. In reality, the browser runs on top of a small Linux operating system.
The choice has been made to not support touch screens, which make KaiOS the ideal OS for affordable phones. Additionally, to the standard functionalities that web browsers offer, KaiOS offers a more advanced Application Programming Interface (API) that allows, for example, to dial a phone number or read SMS messages.
Modern browsers and recent HTML standards already allow to store data and to run a web application locally. KaiOS extends the standard browser API to add additional features useful for apps running on a phone. These include managing Bluetooth, Wi-Fi and mobile connectivity and data storage.
This means that all files and the manifest are stored on a web server. One advantage of hosted apps is that they can be updated on the fly by the host, making sure that all users get the latest version. Alternatively, applications can be packaged in a zip file. Packaged apps are downloaded and installed on the phone. They will load faster than a hosted app and can be used when offline. More importantly, they can be signed and trusted to use more privileged APIs, as explained in the next chapter.

3.1.1 Security model of KaiOS

The security model of KaiOS is described in the online documentation of Kai OS Technologies[3]. KaiOS is made of the following layers (top to bottom):

- **Gaia**: The set of HTML5, CSS, JavaScript apps that are used for operating the phone.
- **Gecko**: The "browser" that runs all apps and offers an extended API to access specific features of the phone, also referred to as the runtime.
- **Gonk**: The Linux kernel and associated software that Gecko runs on.
- **Hardware**: The mobile device that runs KaiOS.

The four layers are shown in Figure 2.
Apps (in Gaia) that want to use features of the phone (in Gonk) cannot do this directly. Gecko acts as a mediator between these layers and only allows access to specific functionalities based on predefined permissions.
Apps are given different levels of privilege depending on their type:

3.1.2 Certified apps

Certified apps are highly trusted. They are approved by the operator or the manufacturer. They are reserved for critical applications like SMS, Bluetooth, camera, system clock, telephony and the default dialer (used to call emergency services).
Certified apps have access to most Web API operations.

3.1.3 Privileged apps

Privileged apps are application that have been reviewed, approved and digitally signed by an authorized KaiStore.
Privileged apps have access to a subset of the Web API that is accessible to certified apps.

3.1.4 Web apps

Web apps refer to all other apps. They are regular web applications that can either be installed (stored locally on the phone) or hosted (stored remotely).
Untrusted apps have access to a subset of Web APIs that contain sufficient security mitigations to be exposed to untrusted web content.
By default, apps have very limited access to Web APIs. If they need more access, the additional API calls must be declared in the manifest of the application. Gecko will only grant access to these calls if the app is sufficiently qualified (certified, trusted or web) for that access.

4 KNOWN SECURITY ISSUES IN KAIOS

4.1 Limited layers and missing defense in depth

KaiOS has fewer software layers than typical smart phone operating systems. In smart phones, each application runs as a different user in a separated runtime. Memory isolation of different processes is done by the memory mapping hardware. Additionally, the operating system prevents accesses to memory or files across different users. To compromise the phone, an application would have to exploit a vulnerability in the runtime, and another one in the operating system.

Figure 1 – Simplified view of Android layers. Each app is executed by a separate runtime running as a different operating system user

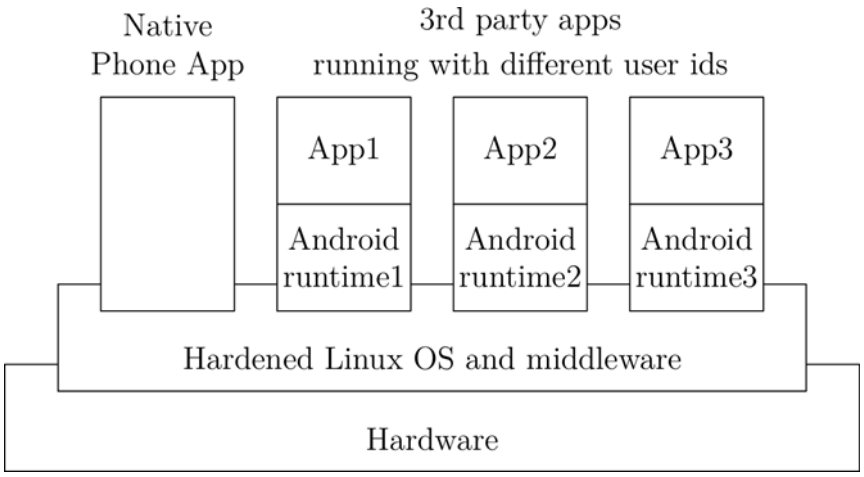
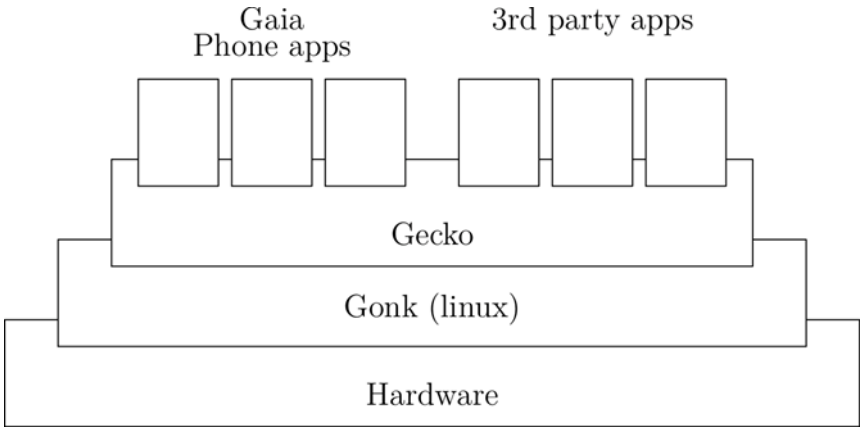


Figure 2 – KaiOS layers. Note that all internal and 3rd party apps are web apps and that they are all run by the Gecko runtime



In KaiOS, all apps are run by the same runtime and thus by the same user. The runtime uses a sandboxing technique to isolate the applications and implements access control. The runtime executes the JavaScript code of all applications. One single bug in the JavaScript interpreter could thus completely compromise all access control and applications. An example for this is the bug discovered in 2019 in the KaiOS browser Gecko 10.05[4]. JavaScript code could be used to crash Gecko and thus force a reboot of the phone. The bug might also have been exploited to run arbitrary code one the phone.

KaiOS has a simple architecture with very few layers. This makes it simple and lightweight but it goes against the principle of defense in depth.

4.2 Rooting and missing root detection

Rooting a phone consists in changing its configuration such that users can get full control of the software on the phone.
Many KaiOS phones have a, relatively, hidden feature that allows debugging of the phone and which can sometimes be used to root the phone. Other KaiOS phones allow copying files to the phone over USB in a way than can be abused to root the phone. However, some more recent phones are known to not be rootable. More details can be found on the web pages of the BananaHackers[5].
One motivation for rooting a KaiOS phone was to install a Whatsapp application when it was not yet officially available on KaiOS.
Rooting is considered dangerous as it could allow controlling and modifying the behaviour of installed applications or spy on their communications. Rooting can be particularly dangerous if it can be achieved by a third party, without the knowledge of the phone's user. In such a scenario, a remote attacker could root the phone and then take control of a financial app, for example.
In smart phones, applications can detect if a phone is rooted, or jailbroken in the case Apple phones. The operating system provides a functionality to detect rooting. On Android this functionality is called SafetyNet. Many financial applications will refuse to run or be installed on a rooted smart phone.
There is no specific API in KaiOS that would allow detecting if a phone was rooted.

4.3 Missing security features

Modern smart phones have two security features that greatly improve the security:

4.3.1 Trusted Execution Environment (TEE)

The TEE, also called secure enclave, is a specific hardware module that can store secrets and keys and execute cryptographic operations with the keys. If an application, or even the operating system, was compromised it would still be virtually impossible to steal the secrets stored in the TEE.
TEE's are also used in smartphones to lock the phone securely. They hold the key that is used to decrypt the phone's content and prevent brute-forcing the PIN code or fingerprint used to unlock the phone. KaiOS phones typically have a four-digit PIN code that can be brute forced in a few hours.
The KaiOS API provides communication channels to secure elements and in 2018 KaiOS has partnered with chip manufacturer that could provide secure elements for KaiOS phones[6]. Although actual KaiOS phones do not seem to have this feature, it is possible that future versions will have it.
Finally, the SIM card is a secure element that is present on all mobile phones. An API for using the SIM card for authentication (mobile ID) existed for FirefoxOS, the precursor of KaiOS. Unfortunately, KaiOS has decided to not support this API.

4.3.2 No fingerprint scanner or face recognition

A known problem with phones is that it is difficult to type long and complex passwords. As a result, users tend to have short and more predictable passwords. To mitigate this, smartphones have different type of biometric scanners (fingerprint, face) that can recognise a user and then unlock the phone or an application, and give access to secrets stored in the trusted execution environment.

4.4 Faulty Software and little incentive to patch

Because of its simple architecture, vulnerabilities in applications can often have a large impact and can sometimes be exploited by a web page containing malicious JavaScript. See for example the report of NCC Group regarding the Alcatel Flip 2 phone[7]. Errors in applications added by the manufacturer allowed to

i. Execute arbitrary commands with root privileges using an undocumented application,
ii. Change the parameters for Over The Air (OTA) updates with JavaScript from any web page,
iii. Disable the PIN of the lock screen by connecting to the phone with a USB cable.

One could argue that KaiOS phones are quite recent and that early smart phones also had their share of critical vulnerabilities. However, the push to make the phones affordable and their simplified architecture make vulnerabilities more probable and more devastating.
The cost argument is illustrated by the fact that Alcatel refused to fix the critical flaws listed above, as the phone model was nearing its end of life. Alcatel's resources were instead directed to fix the issue in newer models. The low price of the phones thus reduces the chances that vulnerabilities will be fixed.

5 ANALYSIS

Having described KaiOS' security model in Chapter 3 and its limitations in Chapter 4, this Chapter analyses the extent to which KaiOS can fulfil the security requirements set forth in Chapter 2.

R 1. Consider the use of strong authentication mechanisms to demonstrate ownership of the device.

Because it lacks biometric sensors, the only way to lock a KaiOS phone is to set a PIN code, which is typically limited to four digits. Without biometry, long PIN codes or passwords, it is not possible to have a strong authentication of the phone's user.
As long as the PIN is not stored in a secure element on the phone, it is also highly probable that physical access to the phone would suffice to bypass the lock screen PIN.
Since the phone includes a SIM card, it contains a strong authentication mechanism to authenticate against the mobile network. The PIN code of the SIM card authenticates the user and cannot be bypassed. Therefore, KaiOS phones can indeed provide a strong authentication mechanism, if the DFS is ran by the Mobile Network Operator (MNO).
It is unfortunate that KaiOS has disabled the API that allowed an application to get an attestation of the phone number from the SIM card (mobileID). This would have allowed other parties than the MNO to strongly authenticate a phone. Thus, now an app can no longer benefit from the strong authentication of the SIM card.

R 2. Make use of hardware and software mechanisms within mobile devices, such as secure elements and TEEs, which can ensure device integrity, and promote the use of devices equipped with security features for use in DFS.

This is currently not the case, but as mentioned in section 4 , secure elements could be added to KaiOS phones in the future.

R 3. Whether an application is designed for deployment on the handset or secure element, it should be designed and implemented in accordance with best practices, including encrypted and authenticated communication and secure coding practices to harden the app.

Encryption, authenticated communication and secure coding practices can be implemented in JavaScript, fulfilling this requirement.

R 4. Apps should be subjected to external security review and penetration testing, and any recommendations acted upon.

The fact that the apps are delivered as a set of HTML, JavaScript and CSS source files gives easy access for security reviews.

R 5. Apps should securely manage username and password information so that adversaries cannot easily forge credentials, and should use strong authentication mechanisms to protect against unauthorized access.

KaiOS does not provide a hardware assisted secure storage of secrets. Therefore, the secrets of all other applications are only one browser bug away.

R 6. Regular security updates are critical to ensure that mobile operating systems running on user devices operate using the latest security patches.

The drive to make feature phones as affordable as possible is in contradiction with the effort to provide regular updates of the operating system.
On the other hand, KaiOS makes it quite simple to provide updates for hosted applications.

R 7. Ensure that security libraries offered by the operating system are correctly designed and implemented and that the cipher suites they support are sufficiently strong.

KaiOS is based on widely used software components (Linux, Gecko). There should not be a significant difference between the security of its libraries and the ones of smart phone operating systems.

R 8. The handset operating system should be configured in a way to reduce the size of the trusted computing base.

The approach of having a monolithic browser acting almost like an operating system and running all applications does not reduce the size of the trusted computing base.

R 9. Consider transitioning away from mobile applications that leverage SMS and USSD in favour of solutions that use strong public key cryptography and end-to-end security.

JavaScript applications in KaiOS can be built with strong public key cryptography. For example, all messages that are exchanged with a server can be signed cryptographically and verified by their recipient. This prevents any manipulation of messages in transit, even by the network operator. It would be even more secure if the keys could be protected by a secure hardware element.

6 CONCLUSIONS

The goal of KaiOS is to provide useful features on an affordable hardware. To be affordable, the hardware has fewer features as compared, for example, to modern smart phones. Security-wise, the biggest lack is the absence of a secure element on the phone that would protect keys and secrets in hardware. The platform is still quite recent, and the missing secure element might become available on newer devices. Risk being related to impact, the current KaiOS phones could already be considered safe enough for transactions of small amounts. For instance, the same type of reasoning has been applied to contactless cards, where small transaction can be done without any pin code.

Another security issue related to cost optimisation is the lack of effort spent to fix security issues in phones that are already on the market. There is hope that lessons will be learned from the first generations of KaiOS phones and the new phones will have fewer issues.

In the current state, KaiOS phones could be a valid platform for low risk DFS that are controlled by the mobile network operator (MNO), as they can leverage the security of the SIM card. With a few improvements on the hardware, newer feature phones could be almost as secure as smart phones for running DFS applications.

Endnotes

1 KaiOS is an operating system for feature phones that is developed by KaiOS Technologies

2 https://www.itu.int/en/ITU-T/studygroups/2017-2020/09/Documents/ITU_FGDFS_SecurityReport.pdf

3 https://developer.kaiostech.com/getting-started/main-concepts/security-access-level

4 https://packetstormsecurity.com/files/151651/Nokia-8810-Denial-Of-Service.html

5 https://sites.google.com/view/bananahackers/root

6 https://www.kaiostech.com/mwc-2018-kai-announces-partnerships-with-mobile-industry-giants/

7 https://www.nccgroup.com/us/our-research/technical-advisory-multiple-vulnerabilities-in-alcatel-flip-2

International Telecommunication Union
Place des Nations
CH-1211 Geneva 20
Switzerland