

FireID

Emerging Mobile Threats White
Paper

October 2010

CONFIDENTIAL



TABLE OF CONTENTS

1. EMERGING MOBILE THREATS AND FIREID.....	2
1.1 ZEUS OVERVIEW	2
1.2 THE MOBILE APPLICATION SANDBOX	2
1.3 MOBILE MALWARE MITIGATORS	3
1.3.1 Storage Isolation	3
1.3.2 Inter-process Communication	3
1.3.3 Sandboxed Runtime Environments	3
1.3.4 Reduced Attack Surface	3
1.3.5 Device Fragmentation.....	3
1.3.6 Application and Device Updates	4
1.4 CONCLUSIONS	4

FireID White Paper

INTRODUCTION

FireID is a highly secure, universal authentication system that makes Internet transactions and accessing secure data safer than ever. End-users and consumers use a small, simple, and user-friendly application on their mobile phones to securely authenticate to any system which typically requires a username and password.

The FireID application does this by generating offline event-synchronous one-time-passwords (OTP) using a locally stored on-device OTP “token” specific to each individual system that requires user authentication. Each token has a unique and separate token Shared Secret that is specific to each system. This is used by FireID to generate OTPs in combination with a secure algorithm to produce a sufficiently unpredictable sequence of random passwords.

Since the FireID mobile soft token runs on mobile devices, the attack surfaces exposed to malware is of critical importance to FireID. Typically researchers have recently discovered that the Zeus botnet has expanded into the mobile malware space. A reasonable assumption is that this malware is targeted at compromising SMS-based two-factor authentication solutions currently in use.

This document examines why the mobile application model is fundamentally more resistant to such malware than the traditional single-user desktop security paradigm.

1. Emerging Mobile Threats and FireID

1.1 Zeus Overview

The attackers use social engineering and browser exploits to trick users into installing a malicious application on their mobile phones. This mobile Trojan is capable of intercepting incoming SMS messages. Each of these messages are inspected and either passed through to the underlying mobile platform or silently acted upon. Regardless, the user is unaware of the existence of the malware.

Since the Trojan is a normal mobile application, it can perform any of the normal application functions as exposed by the mobile platform. Depending on the platform, these functions include, but are not limited to, reading contact information, SMS history, reading and sending SMS messages and executing HTTP calls.

1.2 The Mobile Application Sandbox

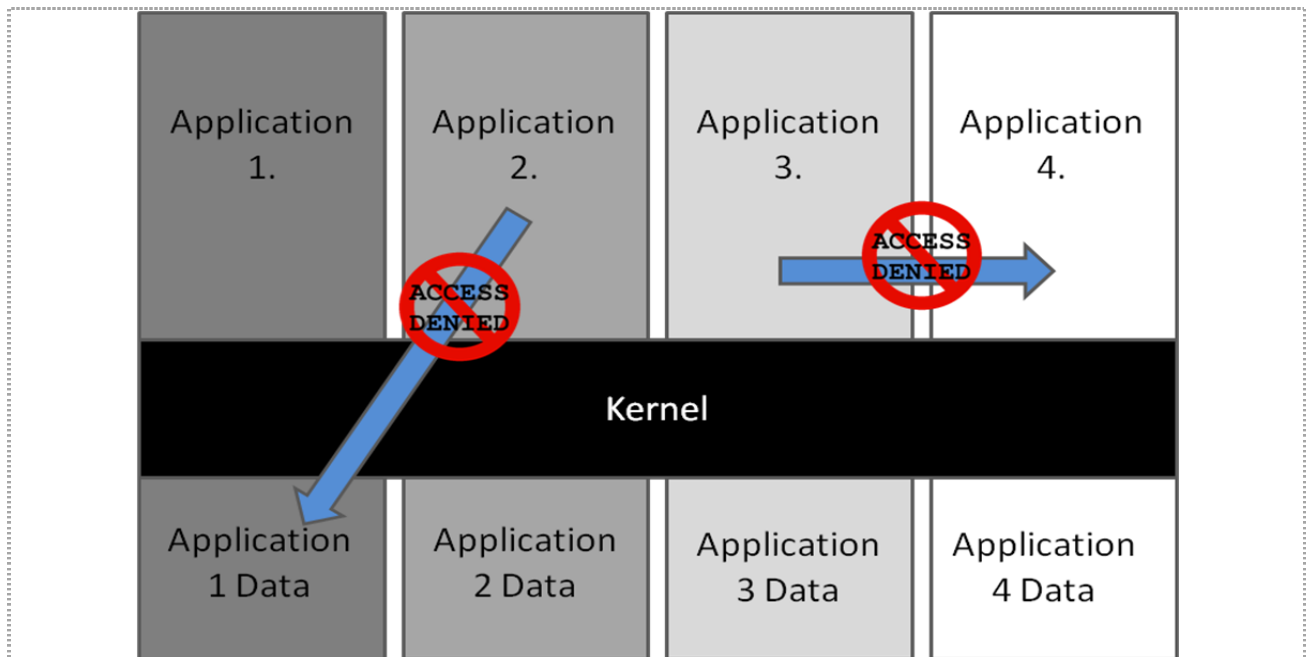


Figure 1 - The Mobile Sandbox Model

In the traditional single-user desktop security paradigm security is generally enforced on a per-user level. While access to protected operating system functionality, for example, is limited to users with elevated permissions, applications can generally access whatever resources the user can access.

In contrast, mobile applications enforce additional security on a per-application basis. In practice, this means that mobile applications are restricted from directly accessing or interacting with other applications.

Mobile platforms are designed to prevent rogue applications from interfering with normal device operation. These platforms only execute applications in a sandboxed environment. Applications can only access the rest of the system using carefully vetted functionality that is explicitly exposed by the mobile operating system.

The exact feature set and security permission granularity exposed to applications vary greatly between the different mobile platforms. For example, Java ME applications cannot intercept normal text SMS messages. Furthermore, the Java ME specification requires that the user must always be prompted before an application is allowed to perform an action that might incur network costs, such as an HTTP request.

More advanced 'smartphone' platforms generally require a form of application signing, but allow access to otherwise restricted functionality. Malicious applications can therefore obtain access to such restricted functionality as explicitly exposed by the mobile platform.

1.3 Mobile Malware Mitigators

1.3.1 Storage Isolation

The FireID mobile application storage, including both the application itself as well as the per-token encrypted shared secret, are isolated from other applications running on the device. The mobile platform either exposes no mechanism to access another application's storage, or explicitly prevents it.

It is important to note that solely compromising the encrypted shared secret does not allow a malicious application to generate OTPs. Since there are many potentially correct PIN codes, an attacker can only verify whether a brute-forced PIN code is correct is by verifying a generated OTP.

1.3.2 Inter-process Communication

Some mobile platforms (e.g. Android) allow and encourage inter-process communication. However, the FireID application explicitly does not share its data with or expose functionality to other applications. Malicious software therefore cannot use the provided communication mechanisms to extract information from the FireID application directly.

1.3.3 Sandboxed Runtime Environments

The sandboxed environments prevent applications from accessing other applications' volatile memory. This prevents malicious applications from reading the unencrypted secret key or PIN code from the FireID application's memory.

1.3.4 Reduced Attack Surface

A common attack on desktop platforms is to install a key logger that captures user keyboard input. This should prove to be a more difficult proposition on mobile devices.

Some platforms (such as Java ME and BlackBerry) are largely immune to such attacks due to the limited nature of the execution sandboxes. Applications advertising such capabilities on these platforms are using the exposed operating system functionality, for example to read SMS history.

The iPhone Marketplace is unlikely to distribute such a malicious application for a significant period. The lack of true multitasking on the iPhone should further deter potential attacks.

Windows Phone 7 applications run in limited sandboxes and do not allow multitasking, native code or non-marketplace applications. It is therefore likely to be a difficult target.

1.3.5 Device Fragmentation

A significant challenge facing mobile application development today is that of device fragmentation. Different device vendors implementing the same platform frequently differ in how they interpret the fine

(and sometimes not-so-fine) details. Certain vendors even have noticeable variance between different phone and firmware versions.

Targeting a specific vulnerability on a specific platform or operating system version is possible. However, no single device or operating system version has an overwhelming market share, so such targeted attacks have limited reach.

FireID can certainly attest to the challenges caused by device fragmentation. Creating a legitimate application that only uses the devices and platforms as intended have proved to require extensive trail-and-error-based testing on physical devices. It is expected that creating malicious applications that exploits targeted platform specific unintended behaviour should be brittle and difficult to scale across many devices.

1.3.6 Application and Device Updates

The mobile application as well as smartphones can be updated to address potential security issues. Attacks should be therefore only be viable for a limited period.

1.4 Conclusions

Mobile platforms are explicitly designed to allow applications access to SMS messages. Applications, both benign and malicious, can therefore use the standard provided platform functionality to obtain some level of access to these messages.

Conversely, mobile platforms are explicitly designed to allow applications to run securely and in isolation, making the FireID solution inherently more secure.

WWW.FIREID.COM

INFO@FIREID.COM

0861 FIRE ID (3473 43)

2ND FLOOR

BLOCK C, OCTO PLACE

ELECTRON ROAD

TECHNOPARK

STELLENBOSCH 7600

WESTERN CAPE

SOUTH AFRICA