

Mobile Device Security



Mobile Device Security

The number of Mobile Computers is growing

Over 1 Billion Android devices have been activated since 2008

800 Million iOS (iPods, iPhones and iPads) devices have been sold

41 Million Blackberry smartphone users world wide

3.5 new threats created per second, including malicious URLs and phishing sites that can be accessed by mobile web browsers.

The average smart phone user installed 35 apps on their device

- Android market place has over 1 Billion apps available
- Apple Appstore has over 1.1 Billion apps available
- Blackberry market place has over 130,000 apps available

33% of Facebook users access the site using mobile devices.

65% of consumers use their smartphone for both business and personal use

28% of U.S. adults use their smartphones and tablets to conduct banking transactions

60% said access to mobile banking was either "important" or "very important" when choosing a bank

Mobile Device Security

Mobile Device Operating Systems

iOS is based on Apple's OS X operating system
Android is based on Linux (v5.0 SE Linux)

Each employ elaborate security models that are designed into their core implementations.

The goals of their creators: to make the platforms inherently secure rather than to force users to rely upon third-party security software.

Each platform's security model protects against today's major threats, including:

- Web-based and network-based attacks.
- Malware
- Social engineering attacks.
- Resource and service availability abuse.
- Malicious and unintentional data loss.
- Attacks on the integrity of the device's data.

Mobile Device Security

Mobile Device Operating Systems

The designers of iOS and Android based their security implementations, to varying degrees, upon five distinct concepts:

Traditional Access Control:

- Traditional access control seeks to protect devices using techniques such as passwords and idle-time screen locking.

Application Signing:

- Each application is stamped with the identity of its author and then made tamper resistant (using a digital signature).

This enables a user to decide whether or not to use an application based on the identity of its author.

In some implementations, a publisher (appstore or marketplace) may also analyze the application for security risks before publication, further increasing the trust in an app.

Encryption: Encryption seeks to conceal data at rest on the device to address the risk from device loss or theft.

Isolation: Isolation techniques attempt to limit an application's ability to access the sensitive data or systems on a device.

Mobile Device Security

Mobile Device Operating Systems

The designers of iOS and Android based their security implementations, to varying degrees, upon five distinct concepts:

Permission based access control:

Permission-based access control grants a set of permissions to each application and then limits each application to accessing device data/systems that are within the scope of those permissions, blocking the applications if they attempt to perform actions that exceed these permissions.

Mobile Device Security

iOS Operating System

iOS leverages all five of the security concepts, its security model is primarily based on four of the five:

- Traditional access control
- Application signing
- Encryption
- Isolation

Traditional Access Control

iOS provides traditional access control security options, including password configuration options as well as account lockout options, remote device wipe and device tracking.

For example, an administrator may choose the strength of the passcode and specify how frequently the user must update their passcode. They can also specify such items as the maximum number of failed login attempts before the device wipes itself.

Mobile Device Security

iOS Operating System

Application Signing

Before software developers can release software to iPhone, iPod, and iPad users, they must go through a registration process with Apple and pay an annual licensing fee. Developers must then “digitally sign” each app with an Apple-issued digital certificate before its release.

This signing process embeds the developer’s identity directly into the app guarantees that the app author is an Apple-approved developer (since only these developers are issued such a certificate), and ensures that the app’s logic cannot be tampered with after its creation by the author.

Apple gives developers two different ways to distribute their applications to customers.

First, anyone wishing to sell their iOS app to the general public must do so by publishing their app on Apple’s App Store. To post an app on the App Store, the software developer must first submit the app for certification by Apple—this certification process typically takes one to two weeks. Once an app has been certified, Apple posts it for sale on its App Store. Free apps go through the same process.

Mobile Device Security

iOS Operating System

Application Signing

Second, corporations wishing to deploy privately-developed apps to their internal workforce may register with Apple's iOS Developer Enterprise program. To be approved for this program, Apple requires that the applicant corporation be certified by Dun and Bradstreet, indicating that they're an established corporation with a clean track record. As a member of this program, enterprises may distribute apps developed in-house via an internal corporate website or by pushing the app using Apple's iOS management platform.

As before, each app must be digitally signed by the enterprise before distribution to the internal workforce. Moreover, internally developed apps can only be used on devices on which the enterprise has installed a digital certificate called a "provisioning profile". If the certificate is ever removed from the device or expires, then all apps signed with the certificate will cease to function.

While Apple explicitly permits corporations to distribute internal applications to their workforces, they prohibit sale/distribution of internally developed apps to third parties.

Apple presumably can simply issue a global revocation for the corporation's provisioning profile, immediately disabling all apps released by the vendor. This certificate requirement also enables a corporation to instantly disable its internally developed applications by simply removing the certificate from a device.

Mobile Device Security

iOS Operating System

Application Signing

It is possible that a malware author could use a stolen identity to register for an account to sell malicious apps on the Apple App Store. Second, Apple does not discuss its app certification approach and it is possible that an attacker could slip malware past this certification process.

Apple has the ability to rapidly remove apps (that are found to be malicious or that violate their licensing agreement) from their App Store, but does not yet appear to possess an automated mechanism to remove malicious apps directly from iPhones/iPads once an app has been installed on the device.

Mobile Device Security

iOS Operating System

Encryption

The latest iPhones, iPads, and iPod Touch devices (that is, those using the iOS 4 operating system and beyond) employ a hybrid encryption model.

First, iOS uses hardware-accelerated AES-256 encryption to encrypt all data stored in the flash memory of the device.

Second, iOS protects specific additional data items, such as email, using an additional layer of encryption.

At first glance, iOS' s full-device encryption approach would appear to offer a high degree of protection. However, since iOS runs background applications even when the user is not logged in to their device, and since these background applications need to access the device' s storage, iOS needs to keep a copy of the decryption key around at all times so it can decrypt the device' s data and provide it to these background apps.

In other words, the majority of the data on each device is encrypted in such a manner that it can be decrypted without the need for the user to input the device' s master passcode.

An attacker with physical access to an iOS device and with a functional jailbreak attack can potentially read most of the device' s data without knowing the device' s passcode.

Mobile Device Security

iOS Operating System

Encryption

A small subset of iOS' s data is secondarily encrypted in such a way that it may only be accessed if the device is unlocked via the user passcode.

If the attacker doesn't have access to the device' s passcode, then this data is essentially 100 percent secure while the device is locked, whether or not an attacker has physical access to the device.

iOS encrypts emails and attachments using this secondary level of encryption. (In April 2014, a security researcher discovered that Apple had turned off this encryption at some point)

Third-party applications can also manually leverage this encryption if they implement the required programming logic.

iOS' s device-level encryption allows rapid device wiping. Since every byte of data on the device is hardware encrypted with an encryption key, a device can be wiped by simply throwing away this key.

iOS devices can be configured to automatically throw away their hardware encryption key if the user enters an incorrect passcode too many times, rendering the data wholly unreadable.

Mobile Device Security

iOS Operating System

Encryption

An attacker that has physical access to a device and a functional jailbreaking tool can potentially obtain access to the data on the device:

Whether or not an iOS device is locked and in the user's pocket, or unlocked in their hand, apps running on the device may freely access iOS's calendar, contact list, photos (many of which are tagged with GPS coordinates), etc., since Apple's hardware decrypts this data on behalf of every running app.

A malicious app could bypass Apple's vetting process, or should an attacker compromise a legitimate app on the device (for example, by using a Web-based attack to compromise the Safari Web browser), the attacker could easily access and steal data from many of the device's systems.

http://tweakimg.net/files/upload/sc_iPhone%20Passwords_tcm502-80443.pdf

Decryption is possible since on current iOS devices the required cryptographic key does not depend on the user's secret passcode. Instead the required key material is completely created from data available within the device and therefore is also in the possession of a possible attacker.

iOS 7 improvements: http://images.apple.com/ipad/business/docs/iOS_Security_Feb14.pdf page 12

Mobile Device Security

iOS Operating System

iOS 8 enhancements

iOS app extensions – More than just third-party keyboards

App extensions allow third-party apps to communicate with each other.

Third-party apps can send and receive data from one app to another through an iOS broker.

This allows social networking apps or seamlessly integrated password managers.

Due to memory limitations, the extensions are expected to only run for a short time and may be cleared from the memory afterwards. The extensions will be executed by the system framework in their own context, meaning that they will not run inside the third-party application's space.

iOS developers who want to use this feature will have to open their app and prepare them to receive communication from app extensions. The iOS broker process will most likely check app extensions' communications to ensure that they are not malicious.

App extensions may technically be able to read other apps' data by using the same default storage, intercept some of the traffic passed between apps or generate a keylogger, the chances of a malicious extension making it to the Apple App Store are quite slim.

The extensions will be pre-screened by Apple, like with all other iOS apps, so malicious extensions will hopefully be stopped before they are distributed to iOS device owners.

Mobile Device Security

iOS Operating System

iOS 8 enhancements

iOS 8 added capabilities to Touch ID, which should help users better secure how they log into third-party applications.

Touch ID allows people to use their fingerprint to gain access to their third-party app passwords which are securely stored in the iOS keychain.

This means that users don't need to enter their password each time they log into an app.

Users can register fingerprints of friends and family to allow them to access devices

Mobile Device Security

iOS Operating System

iOS 8 enhancements

Always-on VPNs: This feature should greatly improve security when an iOS device is connected to a Wi-Fi hotspot, as users do not have to explicitly connect to the VPN in order to benefit from it.

Anti-tracking feature: The introduction of randomized media access control (MAC) addresses when scanning for Wi-Fi networks will make it harder for attackers to track users. Unfortunately, it will still be possible to track these devices once they're connected to the same Wi-Fi network as an attacker or through other identifiers.

MAC address randomization is hard to use because you have to turn off cellular data and location sharing to activate it.

Mobile Device Security

iOS Operating System

iOS 8 enhancements

Expanded data protection

In addition to Mail and third-party apps, the Calendar, Contacts, Reminders, Notes, and Messages apps are protected with a passcode until after the device is unlocked following a reboot.

S/MIME controls per message

S/MIME users can choose to sign and encrypt individual messages for greater control over the security of mail messages.

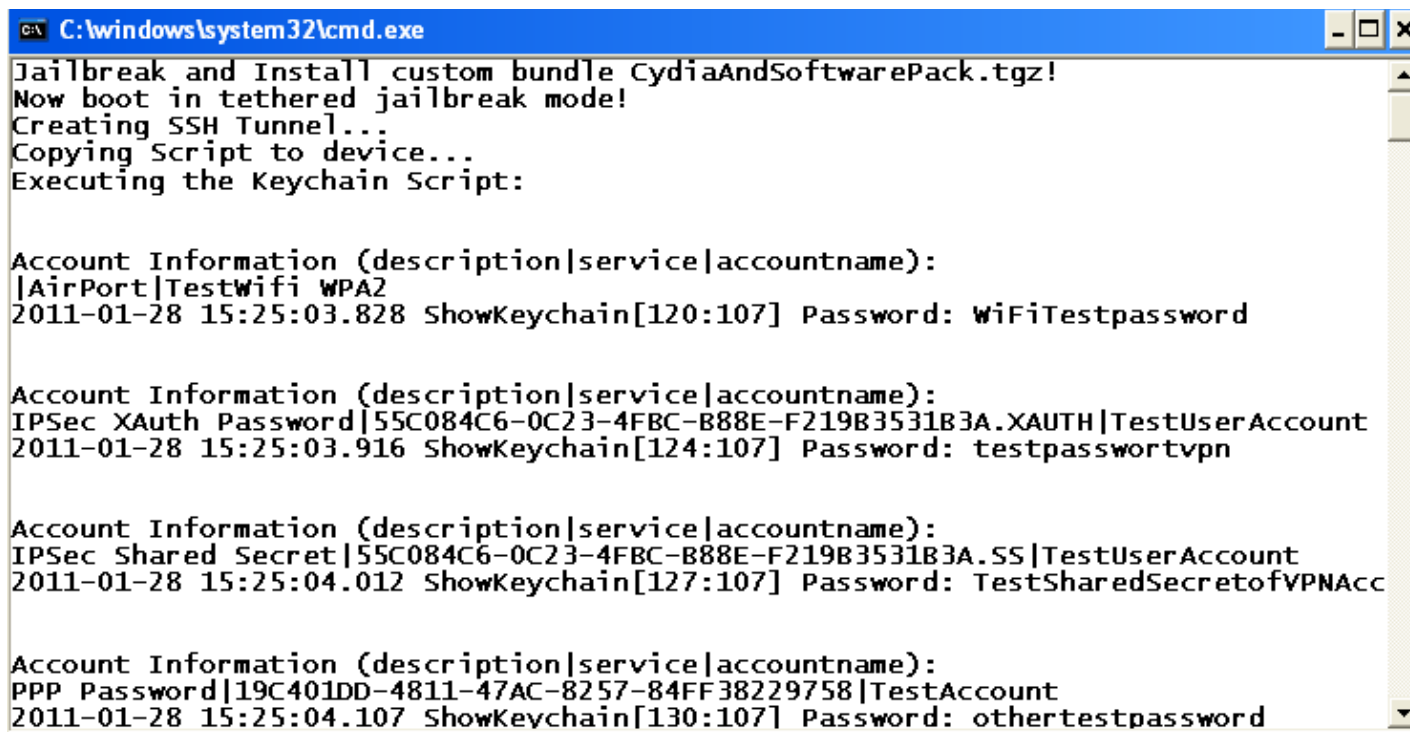
In a document ([PDF link](#)) meant to guide law enforcement officers in requesting user information, *Apple notes that it no longer stores encryption keys for devices with iOS 8*, meaning agencies are unable to gain access even with a valid search warrant. This includes data store on a physical device protected by a passcode, including photos, call history, contacts and more

Mobile Device Security

iOS Operating System

Bypassing Encryption

An attacker that has physical access to a device and a functional jailbreaking tool can potentially obtain access to the data on the device:



```
C:\windows\system32\cmd.exe
Jailbreak and Install custom bundle CydiaAndSoftwarePack.tgz!
Now boot in tethered jailbreak mode!
Creating SSH Tunnel...
Copying Script to device...
Executing the Keychain Script:

Account Information (description|service|accountname):
|AirPort|TestWifi WPA2
2011-01-28 15:25:03.828 ShowKeychain[120:107] Password: WiFiTestpassword

Account Information (description|service|accountname):
IPSec XAuth Password|55C084C6-0C23-4FBC-B88E-F219B3531B3A.XAUTH|TestUserAccount
2011-01-28 15:25:03.916 ShowKeychain[124:107] Password: testpasswordvpn

Account Information (description|service|accountname):
IPSec Shared Secret|55C084C6-0C23-4FBC-B88E-F219B3531B3A.SS|TestUserAccount
2011-01-28 15:25:04.012 ShowKeychain[127:107] Password: TestSharedSecretofVPNacc

Account Information (description|service|accountname):
PPP Password|19C401DD-4811-47AC-8257-84FF38229758|TestAccount
2011-01-28 15:25:04.107 ShowKeychain[130:107] Password: othertestpassword
```

Decryption is possible since on current iOS devices the required cryptographic key does not depend on the user's secret passcode. Instead the required key material is completely created from data available within the device and therefore is also in the possession of a possible attacker.

Mobile Device Security

iOS Operating System

Isolation (Sandboxing)

The iOS operating system isolates each app from every other app on the system
Apps are not allowed to view or modify each other's data, logic, etc.

One app can't even find out if another app is present on the device.

Nor can apps access the iOS operating system kernel, install drivers or obtain root-level (administrator) access to the device.

All third-party applications running on iOS run with the same limited level of device control

Every third-party application running on an iOS device is subject to termination if the device is running low on available memory

The user may also terminate any app at any time with a few taps of the touchscreen.

In addition applications are isolated from the phone's SMS, email in/out-boxes and email attachments within these mailboxes.

Apps are also prohibited from sending SMS messages (without user participation) and from initiating or answering phone calls without the user's participation.

Mobile Device Security

iOS Operating System

Isolation (Sandboxing)

iOS apps are allowed to freely access the following system-level resources without any explicit granting of permission by the user:

- Communicate to any computer over the wireless Internet.
- Access the device's address book including mailing addresses, notes associated with each contact, etc.
- Access the device's calendar entries.
- Access the device's unique identifier (a proprietary ID issued to each device by Apple).
- Access the device's phone number (this may be disabled via a simple configuration change by the user).
- Access the device's music/video files and its photo gallery.
- Access the recent safari search history.
- Access items in the device's auto-completion history.
- Access recently viewed items in the YouTube application.
- Access the Wi-Fi connection logs.

Mobile Device Security

iOS Operating System

Apps can record audio or video and transmit it to their servers if they have permission

What about the security of devices like the Square or similar credit card readers that use the audio jack?

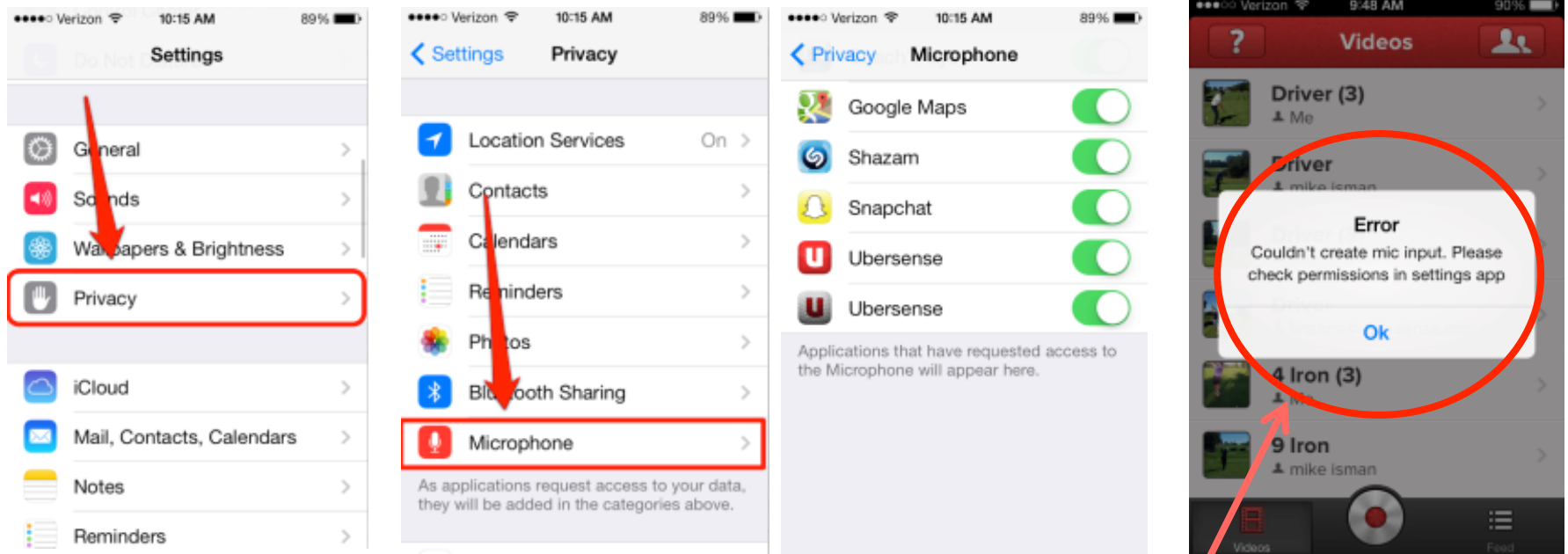


Mobile Device Security

iOS Operating System

Apps can record audio or video and transmit it to their servers if they have permission

In iOS 7 and 8, Apple changed the permissions model to require Apps get specific permissions
To use the microphone input



Error if permission denied

Mobile Device Security

iOS Operating System

Application isolation prevents a number of different attacks, including preventing Web-based and network-based attacks, limiting the impact of malware, preventing malicious data loss, preventing attacks on the integrity of the device's data, and ensuring the availability of the device's services and data.

iOS isolates each app from every other app on the system, this means that if an attacker compromises an app, they will not be able to attack other apps or the iOS operating system itself (unless an unpatched vulnerability in iOS is attacked).

If an attacker were to deliver an attack via a malicious Web page, once malware was running in the Web browser process, it could still access system-wide resources such as the calendar, the contact list, photos, the device's unique ID, etc., since these resources are available for access by all apps under the default iOS isolation policy.

The malicious code could then exfiltrate this sensitive data to the attacker without his code having to ever escape the confines of the browser's sandbox.

Also, resident malicious code within the browser process can also steal any data hosted in or that flows through the browser process itself, including Web passwords, credit card numbers, CCV security codes, account numbers, browsing history, bookmarks, etc. And such a malicious agent in the browser could also initiate malicious transactions on behalf of the user, without their consent.

Mobile Device Security

iOS Operating System

Sandboxing limits the impact of malware

iOS' s isolation framework is effective at preventing classic malware attacks on the iPhone.

Apps can' t access or modify other apps on the system, this prevents a malicious app from infecting or maliciously modifying other apps on the system, as a traditional parasitic computer virus might do.

Sandboxing prevents apps from installing operating system kernel drivers (such as kernel-based malware or root-kits) capable of running with the same administrator-level access as the operating system' s kernel.

Preventing Resource Abuse

iOS' s isolation system can prevent a subset of resource abuse attacks.

iOS apps are given unrestricted access to the Internet, so technically they could be used to launch email-based spam campaigns, search engine optimization campaigns (the attacker tricks a search engine into raising the ranking/visibility of a particular website) and some types of denial of service attacks against websites or a carrier' s network.

On the positive side, iOS' s isolation system prevents the automated transmission of SMS messages or automated initiation of phone calls.

This prevents SMS-based DoS attacks, telephony-based DoS attacks, and SMS-based SPAM attacks on non-jailbroken devices.

Mobile Device Security

iOS Operating System

Preventing Malicious Data Loss

iOS prevents each app from accessing other apps' data—this policy is enforced regardless of whether apps encrypt their data or not, so long as the device has not been jailbroken.

Beyond the library of media files, the calendar, and the contact database which are all accessible to any app, iOS has no centralized repository of shared data that might pose a serious compromise risk.

However, the calendar, media library, etc. often store sensitive information, such as:

- Conference call numbers and passwords.
- Passwords for other systems (for example, bank accounts or enterprise logins).
- Credit card or bank account numbers that might be easily forgotten.
- Key codes for alarms and secure corporate offices.
- Employee names and phone numbers.
- Possibly sensitive audio or video content
- All of these items can be obtained by any third-party app and exfiltrated off the device over the Internet without any warning from the iOS' s security systems.

Mobile Device Security

iOS Operating System

Preventing Attacks on the Integrity of the Device's Data

iOS' s isolation policy allows apps to modify or delete the contents of the calendar and the contact list, but completely prevents modification or deletion of content from the user' s media and photo libraries and from other device systems.

A malicious app could easily delete or modify all of the user' s contacts and calendar entries, but these can easily be recovered from a local backup (automatically created by iTunes during local syncs) or by synchronizing with a cloud-based data source like Exchange, MobileMe, or Google Calendar.

Mobile Device Security

iOS Operating System

Permissions-based Access Control

Apple has built a relatively limited permission system into iOS.

Four system resources that apps may access that first require explicit permission from the user

All other access to system services or data is either explicitly allowed or blocked by iOS' s built-in isolation policy. Here are the permissions that an app may request:

- To access location data from the device' s global positioning system.
- To receive remote notification alerts from the Internet (used by cloud-based services to send real-time notifications to apps running on a user' s iPhone or iPad).
- To initiate an outgoing phone call.
- To send an outgoing SMS or email message.

If an app attempts to use any of these features, the user will first be prompted for permission before the activity is allowed.

If the user grants permission to either the GPS system or the notification alert system, then the app is permanently granted access to these systems.

In contrast, the user is prompted every time an app attempts to initiate an outgoing call or send an SMS message.

Mobile Device Security

iOS Operating System

Vulnerabilities

Security researchers have discovered roughly 300 different vulnerabilities in various versions of the iOS operating system since its initial release.

Most of these vulnerabilities were of lower severity, allowing an attacker to take control of a single process (for example, the Safari process) but not permitting the attacker to take administrator-level control of the device.

The remaining vulnerabilities were of the highest severity, and when exploited, enabled an attacker to take administrator-level control of the device, granting them access to virtually all data and services on the device.

These more severe vulnerabilities are classified as privilege escalation vulnerabilities because they enable an attacker to escalate their privileges and gain total control over the device.

It appears that most of these were developed for the purpose of jail-breaking rather than as a means to maliciously compromise devices.

Apple took an average of 12 days to patch each vulnerability once it was discovered.

<https://www.yahoo.com/tech/you-need-to-update-your-iphone-right-now-heres-how-77621183150.html>

Apple patches iOS 7 for SSL flaws – February 2014 Apple fixed the ‘gotofail’ bug

Mobile Device Security

iOS Operating System

Vulnerabilities

Apple patches iOS 7 for SSL flaws – February 2014 Apple fixed the ‘gotofail’ bug

An attacker could capture or modify data transferred with Mobile Safari, Mail, iCloud and other Apple-created applications even though the communication streams were supposed to be securely encrypted.

Apple iOS prior to 7.0.6 did not properly validate the certificate chain of trust.

Mobile Device Security

iOS Operating System

Vulnerabilities fixed in iOS 8

<http://www.zdnet.com/ios-8-fixes-dozens-of-security-flaws-7000033800/>

With the release of iOS 8, Apple disclosed 53 vulnerabilities that were fixed in the new version.

These vulnerabilities require the ability to execute code on the device, but that could be accomplished with one of the many remote code execution vulnerabilities also disclosed.

Many of these are in the Webkit browser engine, meaning that such an attack could be launched if the user visited a malicious web page.

These issues, many of them severe, remain in earlier versions of iOS.

Here are some examples:

The ability for a rogue access point to steal iOS Wi-Fi credentials using an old and broken authentication protocol which was on by default in iOS. The protocol (LEAP) is disabled by default in iOS 8.

An attacker could have impersonated a WiFi access point, offered to authenticate with LEAP, broken the MS-CHAPv1 hash, and used the derived credentials to authenticate to the intended access point even if that access point supported stronger authentication methods.

This issue was addressed by disabling LEAP by default.

Mobile Device Security

iOS Operating System

Vulnerabilities fixed in iOS 8

Four of these 53 vulnerabilities date to 2013, some of them serious. CVE-2013-5227, for example, involved two methods by which Safari would send usernames and passwords to the wrong site. It was disclosed by Apple in December 2013, but only fixed in OS X at that time.

- An attacker with write access to /tmp to install unverified apps and elevate privileges:

A race condition existed in App Installation. An attacker with the capability of writing to /tmp may have been able to install an unverified app. This issue was addressed by staging files for installation in another directory.

- Opening a maliciously crafted PDF file may lead to an unexpected application termination or arbitrary code execution

An integer overflow existed in the handling of PDF files. This issue was addressed through improved bounds checking.

- Attackers can access sensitive information such as logs or the user's Apple ID.
- Attackers to determine kernel memory characteristics and bypass protections such as ASLR (Address Space Layout Randomization).

Mobile Device Security

iOS Operating System

Some iOS malware

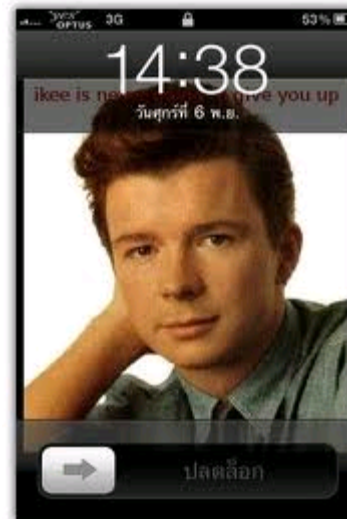
iPhone.Ikee worm Worm (November, 2009):

This computer worm spread over-the-air (for example, across cellular and Wi-Fi networks) to jailbroken iOS devices, changing the device's background wallpaper to display a picture of 80's pop-star Rick Astley.

The worm was only capable of attacking devices that met three criteria:

- 1) The device had to have been previously jailbroken by its owner.
- 2) The owner must have previously installed an SSH (secure shell) application on the device
- 3) The root password had not been changed.

The root password was: alpine



Mobile Device Security

iOS Operating System

Some iOS malware

iPhone.Ikee Worm.B (November, 2009):

This computer worm also spread over-the-air to jailbroken iOS devices using the same SSH default password attack used by iPhoneOS.Ikee.

Once the worm infected a new device, it would lock the screen and display the following text:

“Your iPhone’ s been hacked because it’ s really insecure!
Please visit doiop.com/iHacked and secure your iPhone right now!”

In order to unlock an infected phone, the user was required to pay a €5 ransom to the attacker’ s PayPal account.



Mobile Device Security

iOS Operating System

Some iOS malware

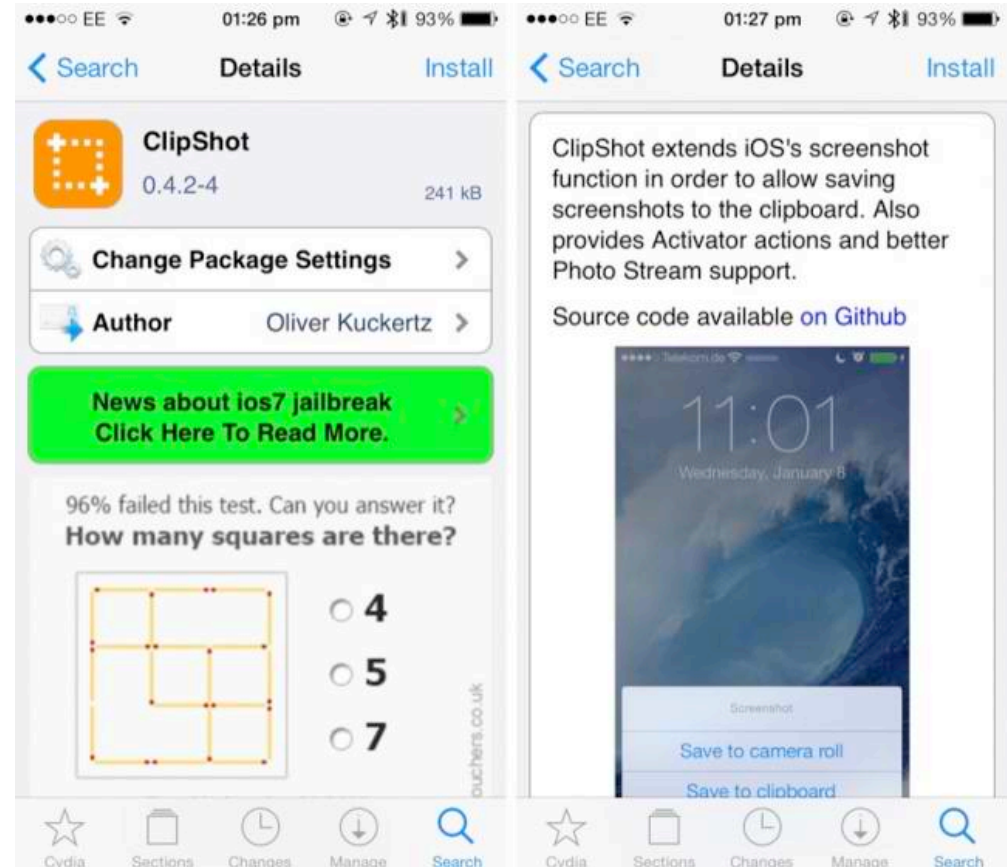
Malware known as AdThief infected 75000 jailbroken iPhones running Cydia

The AdThief malware relied on the Cydia Substrate extension present only on jailbroken Apple devices to hijack advertising bucks.

AdThief malware hooks various advertisement functions and modifies the developer ID (promotion ID) to match that of the attacker, thus the attacker gets paid Ad fees, not the App developer

It targeted 15 mobile advertising kits including Google Mobile Ads and Weibo, four of which were based in the US, two in India and the remainder in China.

<http://www.tripwire.com/state-of-security/top-security-stories/malware-infests-75000-jailbroken-iphones-hijacks-ad-revenue>



Mobile Device Security

iOS Operating System

Some iOS malware

Masque Attack

An iOS app installed using enterprise/ad-hoc provisioning could replace another genuine app installed through the App Store, as long as both apps used the same **bundle identifier**. Ex: com.google.Gmail

This in-house app may display an arbitrary title (like “New Flappy Bird”) that lures the user to install it, but the app can replace another genuine app after installation. All apps can be replaced except iOS preinstalled apps, such as Mobile Safari.

This vulnerability exists because iOS doesn't enforce matching certificates for apps with the same bundle identifier.

An attacker can leverage this vulnerability both through wireless networks and USB.

In demo, an in-house app with a bundle identifier “com.google.Gmail” with a title “New Flappy Bird” was used. The rogue app was signed using an enterprise certificate. When installed, this app, replaced the original Gmail app on the phone and had access to all the cached mail and email password of the original Gmail App.

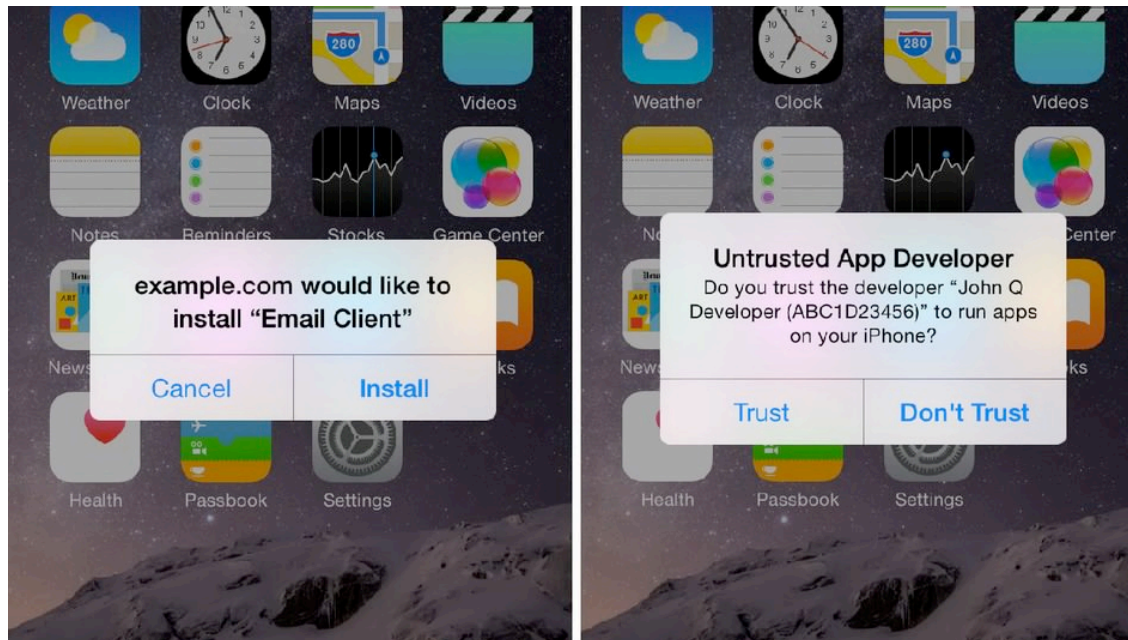
Video Demo: <https://www.youtube.com/watch?v=76ogdpcbIsU>

Mobile Device Security

iOS Operating System

Some iOS malware

Masque Attack



Video Demo: <https://www.youtube.com/watch?v=76ogdpcbIsU>

Mobile Device Security

iOS Operating System

Ways to install pirated iOS apps without jail breaking

First method: Use a cracked Apple enterprise license ([KuaiYong](#)) which allows volume installs of an app to multiple devices. Pirates have found a way to sync a single copy of an enterprise licensed app on 1000s of devices.

Second Method: (Zeusmos) uses a cracked developer certificate to sign an app that has had its DRM stripped off. Once signed, the app can be installed like a developer was testing it.

<http://thenextweb.com/apple/2013/01/01/low-down-dirty-iphone-app-pirates/>

The apps being installed via Kuaiyong may be utilizing bulk enterprise licensing, where the ones distributed by Zeusmos (or other copycat services, which are already springing up) are showing a variety of developer IDs, each of which is being installed multiple times.

Services like Zeusmos have figured out how to ease the process of purchasing a developer slot and using its certificate to install 'cracked' apps, which are widely available on the internet.

Kuaiyong is bypassing Apple's licensing rules to redistribute the same copy of an app over and over.

There are two components to the tool, one which allows for installation of apps directly from your iOS device and the other which is a desktop application that allows install over a cabled connection.

Once you use either, iTunes syncing ceases to work, but that hasn't proved to be too much of a deterrent.

Either method above could allow installing trojaned or malicious software on iOS devices that are not jailbroken.

<http://www.techinasia.com/china-kuaiyong-apple-ios-app-piracy-no-jailbreak/>

<http://www.csoonline.com/article/725183/now-pirated-ios-apps-can-be-installed-without-jailbreak>

Mobile Device Security

Android Operating System

Android

Android combines the Linux operating system and a Java-based platform called Dalvik.

Developers write their apps in the Java programming language and then using Google's Android SDK convert their Java programs to run on the proprietary Dalvik platform on Android devices.

Once converted, such an app can run on any Android device.

Each Android app runs within its own virtual machine (just as Java applications do), and each virtual machine is isolated in its own Linux process.

This model ensures that no process can access the resources of any another process (unless the device is jailbroken).

While Java's virtual machine was designed to be a secure, "sandboxed" system capable of containing potentially malicious programs, Android does not rely upon its virtual machine technology to enforce security.

Instead, all protection is enforced directly by the Linux-based Android operating system.

Mobile Device Security

Android Operating System

Android

Android's security model is primarily based on three of the five security concepts: traditional access control, isolation, and a permission-based security model.

Traditional Access Control

Android 2.X versions provide rudimentary password configuration options, including the ability to specify the strength of the device passcode, specify the phone's lockout time span, and specify how many failed login attempts must occur before the device wipes its data.

Android 3.0 introduced the notion of password expiration, enabling administrators to compel users to update their password on a regular schedule.

Mobile Device Security

Android Operating System

Isolation

Like iOS, Android employs a strong isolation system to ensure that apps only access approved system resources. This isolates each app from other apps on the system, and also prevents apps from accessing or modifying the operating system kernel, ensuring that a malicious app can't gain administrator-level control over a device.

The default isolation policy prohibits access to virtually every subsystem of the device, with the following noteworthy *exceptions*:

- Apps may obtain the list of apps installed on the device and examine each application's programming logic (but not its private data).
- Apps may read (but not write to) the contents of the user's SD flash card, which typically holds the user's music, video files, installed programs, and possibly documents or saved attachments.
- Apps may read all of the data on the SD card without restriction (regardless of which app created a particular piece of data, all apps can read that data).
- Apps may launch other applications on the system, such as the Web browser, the maps application, etc.

Android permits applications to request à-la-carte access to the device's other subsystems—the isolation system then enforces each app's expanded set of permissions.

Mobile Device Security

Android Operating System

Is Isolation Effective?

Web browsers are by far the most targeted class of legitimate application, since attackers know that Web browsers often have security flaws that can easily be exploited by a properly crafted malicious Web page.

If an attacker posted a malicious Web page that attacked a known flaw of Android's Web browser and a user used a vulnerable browser to visit it, the attack could inject itself into the Android browser and begin running.

Once running in the Web browser's process, would this attack pose a threat?

Android's isolation policy would ensure that the attack could not spread beyond the browser to other apps on the system or to the operating system kernel itself.

However, such an attack could access any parts of the system that the Web browser app had been granted permission to access. For example, the Web browser could have permission to save or to modify data on the user's SD storage card.

Mobile Device Security

Android Operating System

Web browser exploits

Also, malicious code within the attacked process can also steal any data that flows through the process itself. In the case of a Web browser, the attack could easily obtain login names, passwords, credit card numbers, CCV security codes, account numbers, browsing history, bookmarks, etc.

Mobile users often access internal enterprise applications via their mobile Web browser, this could lead to leakage of highly sensitive enterprise data, even if VPN or SSL encryption are employed.

And such a malicious agent in the browser could also initiate malicious transactions on behalf of the user, without their consent.

Mobile Device Security

Android Operating System

Is Isolation Effective?

Consider the ability of Android's isolation system to protect against malicious apps such as Trojan horses and spyware.

Since Android's isolation system is designed to isolate each app from other apps on the system, this ensures that a malicious app can't tamper with other apps on the system, access their private data or access the Android operating system kernel.

However, an application can request excessive permissions and if allowed, it could access: the email inbox, the GPS system, the network and more.

This allows many categories of attacks to operate in the isolation mechanisms of Android.

Mobile Device Security

Android Operating System

Permissions-based Access Control

Android applications can do very little without explicitly requesting permission from the user to do so.

If an app wants to communicate over the Internet, at install time, it must explicitly request permission from the user to do this; otherwise the default isolation policy blocks it from initiating direct network communications.

Each Android app contains an embedded list of permissions that it needs in order to function properly.

This list of requests is presented to the user in non-technical language at the time an app is installed on the device, and the user can then decide whether or not to allow the app to be installed based on their tolerance for risk. If the user chooses to proceed with the installation, the app is granted permission to access all of the requested subsystems.

Mobile Device Security

Android Operating System

Permissions-based Access Control

Third-party apps can request permission to use the following high-level subsystems:

Networking subsystems:

- Apps can establish network connections with other networked devices over Wi-Fi or using the cellular signal.

Device identifiers:

- Apps can obtain the device's phone number, the device ID (IMEI) number, its SIM card's serial number, and the device's subscriber ID (IMSI) number.

These codes can be used by criminals to commit cellular phone fraud.

Messaging systems:

- Apps can access emails and attachments in the device's inbox, outbox, and SMS systems.

Apps can also initiate transmission of outgoing emails and SMS messages without user prompting and intercept incoming emails and SMS messages.

Calendar and Address book:

- Apps can read, modify, delete, and add new entries to the system calendar and address book.

Multimedia and image files:

- Apps may access multimedia (for example, MP3 files) and pictures hosted by the device's photo application.

Mobile Device Security

Android Operating System

Permissions-based Access Control

Apps can request to save, modify, or delete existing data on external plug-and-play SD memory cards.

Once granted this permission, apps have unrestricted access to all of the data on the SD card, which is not encrypted by default.

Global positioning system:

- Apps may obtain the device's location.

Telephony system:

- Apps can initiate and potentially terminate phone calls without the user's consent.

Logs and browsing history:

- Apps may access the device's logs (such as the log of outgoing and incoming calls, the system's error log, etc.) as well as the Web browser's list of bookmarks and surfing history.

Task list:

- An app may obtain the list of currently running apps.

Mobile Device Security

Android Operating System

Application Signing

Android's Digital Signing Model

The goal of digitally signing an application is ensure that the app's logic is not tampered with, to allow a user of the app to determine the identity of the app's author.

Google's approach undermines both of these goals.

Like Apple, the Android operating system will only install and run apps that have been properly signed with a digital certificate.

However, unlike Apple, software developers need not apply to Google to obtain a code-signing certificate. Instead, application developers can generate their own signing certificates, as often as they like, without any oversight. The software developer can place any company name and contact information in their certificate that they like.

The result is that a malware author can generate "anonymous" digital certificates as often as they like and none of these certificates or malware signed with them can be traced back to the author.

Mobile Device Security

Android Operating System

Application Signing

In order for developers to sell their apps on Google's official Android Marketplace, developers must pay a \$25 fee via credit card.

This enables Google to associate the payee with the digital certificate used to digitally sign the developer's apps and should act as a mild deterrent against malware authors posting malware on the Android Marketplace (if they use their own credit card).

However, given that developers have the ability to distribute their apps from virtually any website on the Internet—not just the Android marketplace—malware programs can also be distributed with anonymity without any vetting by Google.

This approach makes it easier for attackers to add Trojan horses to existing legitimate apps. The attacker can obtain a legitimate app, add some malicious logic to the app, and then re-sign the updated version with an anonymous certificate and post it onto the Internet.

While the newly signed app will lose its original digital signature, Android will certify and install the newly signed malicious app with its anonymous digital signature.

Android's model does not realistically prevent tampering.

Mobile Device Security

Android Operating System

Application Signing

Like iOS, Android will never silently install an app onto a device. The user is always notified before a new application is installed.

This mostly prevents drive-by attacks common on PCs and requires attackers to employ social engineering (offer for free) to trick users into agreeing to install malicious apps on their devices.

Implementation Problems with the App Signing

The majority of devices running Google's Android operating system are susceptible to hacks that allow malicious apps to bypass a key security sandbox so they can steal user credentials, read e-mail, and access payment histories and other sensitive data

The high-impact vulnerability has existed in Android since the release of version 2.1 in early 2010, researchers from Bluebox Security said.

It grants malicious apps special access to Android resources that are typically off-limits.

Google developers have introduced changes that limit some of the damage that malicious apps can do in Android 4.4

Mobile Device Security

Android Operating System

Implementation Problems with the App Signing

The vulnerability stems from the failure of Android to verify the validity of cryptographic certificates that accompany each app installed on a device.

The OS relies on the credentials when allocating special privileges that allow a handful of apps to bypass Android sandboxing. Under normal conditions, the sandbox prevents programs from accessing data belonging to other apps or to sensitive parts of the OS. Select apps (and Device management extensions), are permitted to break out of the sandbox. Adobe Flash in all but version 4.4, for instance, is permitted to act as a plugin for any other app installed on the phone, presumably to allow it to add animation and graphics support. Similarly, Google Wallet is permitted to access Near Field Communication hardware that processes payment information.

Android fails to verify the chain of certificates used to certify an app belongs to this elite class of super privileged programs. As a result, a maliciously developed app can include an invalid certificate claiming it's Flash, Wallet, or any other app hard coded into Android.

The OS, in turn, will give the rogue app the same special privileges assigned to the legitimate app without ever taking the time to detect the certificate forgery.

Mobile Device Security

Android Operating System

Encryption

The latest generation of Android tablet devices (running Android 3.0, 4.0 and 5.0) support hardware encryption to protect data.

Devices running earlier versions of Android (including virtually all Android-based mobile phones) rely upon the isolation model, instead of encryption, to protect data such as passwords, user names, and application-specific data. Some version 2.3 devices encrypt data using software.

This means that if an attacker is able to jailbreak a device or otherwise obtain administrator-level access to a device by exploiting a vulnerability or by obtaining physical access to a device, they can access virtually every byte of data on the device, including most of the passwords, Exchange/private email account credentials, etc.

Mobile Device Security

Android Operating System

Android 5.0

Full device encryption occurs at first boot, using a unique key that is not stored on the device.

Android 5.0, (Security Enhanced Linux) SELinux Enforcing mode is required for all applications on all devices.

Share your device securely with guest user mode

Create multiple user accounts to enable friends to log in on your device

Android Smart Lock to secure your phone or tablet by pairing it with a trusted device like your wearable or even your car.

Mobile Device Security

Android Operating System

Vulnerabilities

Security researchers have discovered a couple dozen different vulnerabilities in various versions of the Android operating system since its initial release. Most were of lower severity and would only allow an attacker to take control of a single process (for example, the Web browser process) but not permit the attacker to take administrator-level control of the device.

The high severity vulnerabilities, when exploited, enabled an attacker to take root-level control of the device, granting them access to virtually all data on the device.

Many carriers don't release AndroidOS upgrades to their customers. As a result some 50% of devices in use have severe unpatched vulnerabilities.

Google took an average of eight days to patch each vulnerability once it was discovered.

Vulnerabilities Grant Elevated Permissions to Malicious Apps' When Android Updates 3/23/2014

Flaws recently found in Android puts devices running the mobile operating system at risk of privilege elevation attacks.

The newly-defined class of vulnerabilities increase permissions of malicious apps when Android is updated.

<http://www.zdnet.com/android-bugs-leave-every-smartphone-and-tablet-vulnerable-to-privilege-escalation-7000027589/>

<http://www.informatics.indiana.edu/xw7/papers/privilegeescalationthroughandroidupdating.pdf>

Mobile Device Security

Android Operating System

Some Android Malware

AndroidOS.FakePlayer (August, 2010):

This malicious app masquerades as a media player application. Once installed, it silently sends SMS messages (at a cost of several dollars per message) to premium SMS numbers in Russia. Devices connected to wireless carriers outside of Russia are unaffected since the SMS messages are not properly delivered; however, this threat illustrates how easy it is to steal funds from unsuspecting users.

2010: Google removed banking malware that had gathered information on more than 1m Android users – Google marketplace can delete malicious apps installed via Google Play

Android.Rootcager (February, 2011):

Also known as Android.DroidDream.

The attacker infected and redistributed more than 58 legitimate applications on Google's App Market. Hundreds of thousands of users were infected, tricked into thinking they were downloading legitimate applications. Once installed by the user, the threat attempted to exploit two different vulnerabilities in Android *to obtain administrator-level control of the device*.

The threat then installed additional software on the device, without the user's consent. The software exfiltrates a number of confidential items, including: device ID/serial numbers, device model information, carrier information, and has the ability to download and install future malware packages without the user's knowledge (this is only possible since the threat exploited a vulnerability to bypass Android's isolation model).

Both vulnerabilities used by Android.Rootcager were patched in Android's 2.3 release; however, most Android-based devices on the market are running earlier Android versions meaning that most devices are still susceptible to this style of attack.

Mobile Device Security

Android Operating System

Some Android Malware

Android.Bgserv (March, 2011):

In response to the Android.Rootcager threat, Google deployed a tool over-the-air to clean up infected Android devices. Shortly after this cleanup tool was released, attackers capitalized on the hype and released a malicious fake version of the cleanup tool.

This Trojan horse exfiltrates user data such as the devices IMEI number and its phone number to a server in China.

2011: Zeus malware for Android steals financial data

2012: Spam Soldier

Spam Soldier sent spam messages with infected links to contacts in the address book of the infected phone.

Spam sent from the phone and any replies to those messages were hidden from the operator by the malware.

<http://www.csoonline.com/article/729394/sms-becoming-meaty-attraction-for-spammers>

Mobile Device Security

Android Operating System

Android Malware

Android Botnet Infects Up to 1 Million Phones in China

http://threatpost.com/en_us/blogs/new-android-botnet-infects-1m-phones-china-011513

Up to a million Android users in China could be part of a large mobile botnet according to research unveiled by Kingsoft Security, a Hong Kong-based security company, this week.

The botnet has spread across phones running the Android operating system via Android.Troj.mdk, a Trojan that researchers said exists in upwards of 7,000 applications available in the non-Google marketplaces, including the popular Temple Run and Fishing Joy games.

According to reports, the strain of malware was discovered in 2011 but recent analysis has shown the botnet has ramped up infection rates and at this point might have infected one million smartphones.

One report states that the botnet opens phones to remote hijacks and unauthorized purchases. A separate news entity, which first reported about the botnet, claims the malware has caused some phones to randomly open "strange software" that is tricky to remove.

Mobile Device Security

Android Operating System

Android Malware

Trojan SMS malware infect 300,000 devices, net crooks \$6m (13th February 2014)

The trojan reportedly infects users' handsets via a bogus permissions notification, which when agreed to instigates a complex process that forces the victim to send text messages to a premium-rate number owned by the hackers.

<http://www.v3.co.uk/v3-uk/news/2328691/android-apps-with-trojan-sms-malware-infect-300-000-devices-net-crooks-usd6m>

Mobile malware: <http://www.csoonline.com/article/2834571/malware-cybercrime/new-technique-allows-attackers-to-hide-stealthy-android-malware-in-images.html>

Mobile Device Security

Android Operating System

Android Malware

Google Bouncer was introduced in February 2012 to scan all Google marketplace apps for malicious code. Several security experts have tested Bouncer and found that its detection rate is around 25%. <http://www.csc.ncsu.edu/faculty/jiang/appverify/>

Many different ways have been found to bypass Bouncer as it is possible for an app to tell it is running in the Bouncer environment and modify its behavior.

April 2014: Google announces continuous app scanning:

<https://www.yahoo.com/tech/androids-new-security-feature-will-regularly-scan-your-82298930780.html>

Android Botnet Targets Banks in Middle East (April 2, 2014)

A recently detected botnet targets Android users who do business at banks in the Middle East. More than 2,700 Android phones have been infected, and more than 28,000 text messages intercepted.

<http://krebsonsecurity.com/2014/04/android-botnet-targets-middle-east-banks/>

Mobile Device Security

Android Operating System

Android Malware – Bootkit for Android

<http://thehackernews.com/2014/04/most-sophisticated-android-bootkit.html>

Malware called Oldboot.A infected more than 500,000 Smartphone users worldwide with Android operating system, mostly in China.

Oldboot is designed to re-infect Mobile devices even after a thorough cleanup. It resides in the memory of infected devices; It modifies the devices' boot partition and booting script file to launch a system service and extract malicious application during the early stage of system's boot process.

Oldboot.B variant has advanced stealth techniques to defend it against antivirus software, and automatic analysis tools.

Mobile Device Security

Android Operating System

Android Malware – Bootkit for Android

<http://thehackernews.com/2014/04/most-sophisticated-android-bootkit.html>

Oldboot.B, Android Bootkit malware has following capabilities:

- It can install malicious apps silently in the background.
- It can inject malicious modules into system process.
- It can prevent malware apps from uninstalling.
- Oldboot.B can modify the browser's homepage.
- It has ability to uninstall or disable Mobile Antivirus software.

Android bootkit can't be removed with antivirus

<http://www.ehackingnews.com/2014/01/first-android-bootkit-virus-found.html>

Threat gets into the device when user reflash their smartphones with the modified firmware containing this Trojan.

Mobile Device Security

Malware for mobiles can be divided into major categories of:

Data Stealer & Spying Tools - steal passwords, personal information, text messages and even voice calls

Premium Service Abusers - use your phone to make expensive toll calls and TXT messages with per message charges

Click Fraudster - use your smartphone's internet connection to fill out surveys or click on advertising where the virus writer gets paid per survey or clicked ad.

Malicious Downloader - malware that is not itself malicious, but downloads other malware that is.

The following is a list of mobile device operating systems ranked by the amount of malware in the wild for each:

1. Android
2. SymbianOS
3. Windows Mobile
4. Blackberry RIM
5. Apple iOS

The most common way malware gets on a smartphone is through trojan app downloads (including JAVA apps) where the application does something you want, but in the background also has malicious features.

Mobile Device Security

Android Malware Analysis Sandboxes

anubis.iseclab.org

mobilesandbox.org

joesecurity.org – apk-analyzer.net

www.virustotal.com

Linux Distro for mobile forensics, malware analysis, and security testing (Santoku)

<https://santoku-linux.com/download>

<http://damnvulnerableiosapp.com/#downloads> – iOS app for practicing PenTest and app auditing

Artifacts like Web browsing history, deleted files, deleted text messages, call history and downloaded email can be recovered from the file system of mobile phones

Mobile app Audit framework:

<https://www.mwrinfosecurity.com/products/drozer/>

Mobile Device Security

Recent Android Malware

Trojan Mobile Antivirus App:

<http://www.androidauthority.com/armor-for-android-342192/>

Android Ransomware

Scares the victim into believing their phone is overrun with malware and then kills key processes and deletes a significant amount of files in the attempt to prevent the victim from restoring their phone from a backup file.

Finally, after a short period of time the malware presents the victim with a special lock screen where the only option is to pay the attacker \$99.99 via credit card for the “full” version of the antivirus software.

In the case of *Fake Defender*, paying the ransom has no effect: nothing is downloaded and nothing is restored.

The only option for the victim is to wipe the phone and start from scratch.

Mobile Device Security

Recent Android Malware

Botnets of infected mobile devices are active

NotCompatible, the mobile Trojan was discovered in 2012 and was the first Android malware to be distributed as a drive-by download from compromised websites

New version, NotCompatible.C, encrypts its communications with the Command and Control servers (C&C servers)

NotCompatible.C can communicate with other infected devices directly, forming a peer-to-peer network that offers redundancy in case the main C&C servers are shut down

The attackers are using load balancing and geolocation techniques in the infrastructure side so that infected devices are redirected to one of more than 10 separate servers located across Sweden, Poland, the Netherlands, the U.K., and the U.S.

Mobile Device Security

Recent Android Malware

NotCompatible.C botnet has been used to:

- Send spam to MS Live, AOL, Yahoo and Comcast addresses
- Purchase tickets in bulk from Ticketmaster, Live Nation, EventShopper and Craigslist
- Launch brute-force password guessing attacks against WordPress sites
- Control compromised sites through Web shells

Security researchers believe that the botnet is likely rented to other cybercriminals for different activities

Using the NotCompatible proxy, an attacker could potentially attack corporate networks from inside doing anything from enumerating vulnerable hosts inside the network, to exploiting vulnerabilities and searching for exposed data

Mobile Device Security

Recently Patched Android Vulnerability

Android packages (APKs) downloaded through Google Play are installed to a protected space, whereas apps downloaded through a third-party store are saved to unprotected local storage

A trojan malicious app could be written to detect when the compromised user is installing a new app

The malicious app will check whether the new app is being installed through a third-party store or Google Play

If the app is going to be saved in an unprotected space, the malicious app overwrites the legitimate app with malware while a user views a permission page

More permissions could be provided than detailed in the permissions page, and the device becomes officially compromised since the APK integrity is not checked at time of use

Mobile Device Security

Cross Device Infections

2013 also saw a new vector of infection for Windows: through Android.

Android/Claco is disguised as a harmless 'cleaner' application, but when plugged into the victim's Windows PC, Claco attempts to autorun a malicious file it placed in the root of the phone's filesystem.

The beginning of 2014 saw a new piece of Windows malware designed to infect Android devices.

Mobile Device Security

Cross Device Infections

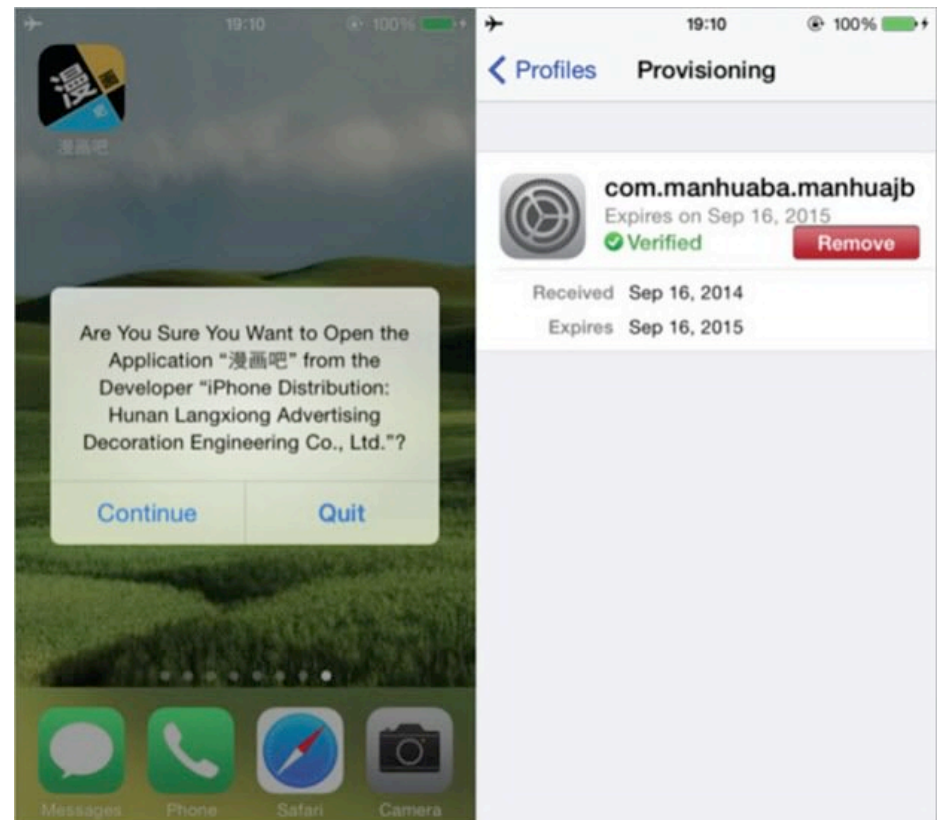
WireLurker Malware (November 6, 2014)

WireLurker infects iOS devices through apps downloaded to OS X machines from unauthorized, 3rd party app stores. The malware then infects iOS devices when they are plugged into infected OS X machines with a USB cable.

The malware is able to install malicious and infected programs on non-jailbroken iOS devices, by using enterprise provisioning techniques, thus appearing to be an in-house application.

Apple has revoked a cryptographic certificate that was being used to sign WireLurker.

So far, the malware appears to be affecting users in China. Apple urges customers to obtain apps only from the legitimate app store.



Mobile Device Security

Table of the top 20 found passwords is shown at the right.

26.83% of all passwords could be guessed by attempting these 20 combinations!

Year of birth is also a common passcode

* <http://www.datagenetics.com/blog/september32012/>

	PIN	Freq
#1	1234	10.713%
#2	1111	6.016%
#3	0000	1.881%
#4	1212	1.197%
#5	7777	0.745%
#6	1004	0.616%
#7	2000	0.613%
#8	4444	0.526%
#9	2222	0.516%
#10	6969	0.512%
#11	9999	0.451%
#12	3333	0.419%
#13	5555	0.395%
#14	6666	0.391%
#15	1122	0.366%
#16	1313	0.304%
#17	8888	0.303%
#18	4321	0.293%
#19	2001	0.290%
#20	1010	0.285%

Mobile Device Security

Apple's iOS7 PRNG Weaker Than Previous Version (March 14 & 16, 2014)

Apple changes its internal pseudorandom number generator (PRNG) with iOS 7 and researchers are saying that it is weaker than the previous version.

The weakness could allow attackers to more easily exploit a vulnerability in the operating system's kernel. Weakens ASLR.

http://www.theregister.co.uk/2014/03/16/ios_7_has_weak_random_number_generator/
<http://www.scmagazine.com/researcher-finds-easier-way-to-exploit-ios-7-kernel-vulnerabilities/article/338390/>

Mobile Device Security

Application Security Report from February 2013

Appauthority.com evaluated the top 100 most popular apps

- The vast majority of free apps send and receive data to outside parties without encryption.
- 96% of total apps share data with advertising networks and/or analytics companies.
- 79% of the top 50 free iOS and Android apps are associated with risky behaviors or privacy issues.

Overall, iOS apps exhibited more risky behaviors than Android apps.

- Entertainment apps were the worst offenders out of the top five categories, with the highest number of apps that track for location and share data with advertising networks and/or analytics companies.
- While 14% of iOS apps had access to a user's calendar, none of the Android apps had similar access.
- More than half of the total apps track for location by accessing the device GPS or using other location tracking methods.
- More than 80% of apps across categories come from different unique, individual developers.

Mobile Device Security

Mobile Apps have weaker security than websites:

<http://www.fireeye.com/blog/technical/2014/02/amazons-mobile-shopping-clients-and-captcha.html>

FireEye mobile security researchers have found security issues within Amazon's mobile apps on both Android and iOS platforms through which attackers can crack the passwords of target Amazon accounts. Amazon confirmed our findings and hot fixed the issue.

Two example security issues within Amazon's mobile apps on both Android and iOS platforms:

- No limitation or CAPTCHA for password attempts
- Weak password policy
- Attackers can exploit these two security issues remotely against target Amazon accounts.

Many re-use their Amazon password for other things, so this would have been a handy place to brute force passwords

Mobile Device Security

Study from Carnegie Mellon University found that smartphones track our location every three minutes on average

Study found that popular Android apps track a user at an average 6,200 times during a two-week period, which is every three minutes

Apps tested included:

- Weather Channel
- Groupon
- Facebook Messenger
- Yelp
- Uber
- Instagram
- Snapchat
- iHeartRadio

Mobile Device Security

Wireless communication can include Geo Location artifacts like:

```
HTTP URI includes browserlocation, wifi=mac:76-e1-b6-00-11-22%7Cssid:TimG
%C3%A2%C2%80%99s%20iPad%7Ccss:-39&wifi=mac:98-2c-
be-44-55-66%7Cssid=ciscod%7Ccss:-75&wifi=mac:c4-3d-
c7-77-88-99%7Cssid:PWNEED100%7Ccss:-87
```

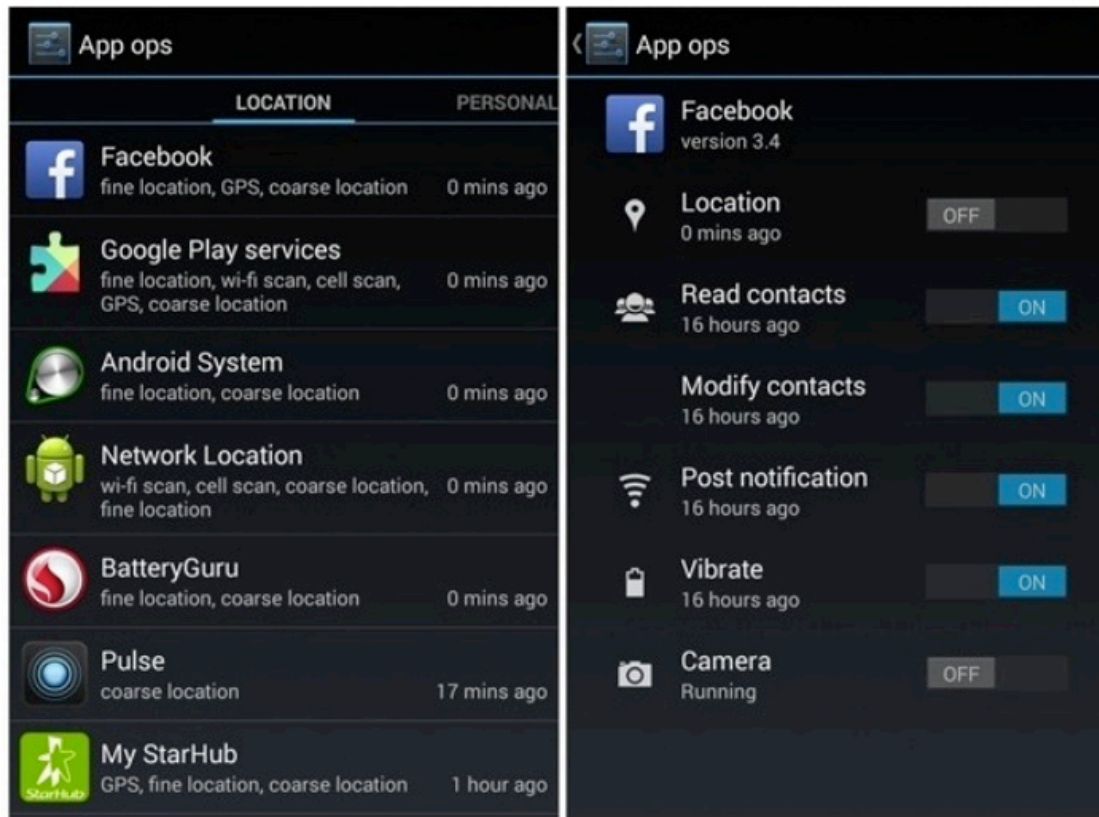
URL shows location of wireless device when connected to maps.googleapis.com/maps/api

ss is signal strength

```
{
  "accuracy" : 30.0,
  "location" : {
    "lat" : 12.7402630
    "lng" : -81.1030548
  },
  "status" : "OK"
}
```

SSID improves accuracy and works even if using SSL or VPN

Mobile Device Security



AppOps show GeoLocation permissions for each App

Mobile Device Security

In Last 12 Months

Files Infected

TROJ_ZAccess (password stealer - 7 variants)	1181
AndroidOS malware (20 variants)	967
TROJ_SpyEye (password stealer - 35 variants)	736
Troj_ZBot (password stealer - 48 variants)	371
BKDR_QAKbot - (17 variants)	191
BKDR_Bifrose (keylogger - 14 variants)	122
Troj_Banker (password stealer - 11 variants)	70
BKDR_Poison (password stealer, RAT - 6 variants)	34
TROJ_Carberb (password stealer - 4 variants)	13
Sinowal (Torpig – password stealer – 2 variant)	11

Only 15% of smart phone users install antivirus software

Securing Your Mobile Device

Antivirus software for Mobiles: Android

LOOKOUT – antivirus scanner and phone locator

AVG antivirus – scans apps and media

Norton Android mobile Security Lite – scans downloaded apps and app updates

NetQin Mobile Security – virus scanning and phone locator

All of these Apps are **free**

