ON THE FEASIBILITY OF LARGE-SCALE INFECTIONS OF IOS DEVICES

Tielei Wang, Yeongjin Jang, Yizheng Chen, Simon Chung, Billy Lau, and Wenke Lee,
Georgia Institute of Technology

Presented by Sai Tej Kancharla

CONTENTS

- Introduction
- iOS security
- JEKYLL on iOS
- Drawbacks of malicious apps
- Ways of Attack
- Measurement
- Prevention
- Conclusion

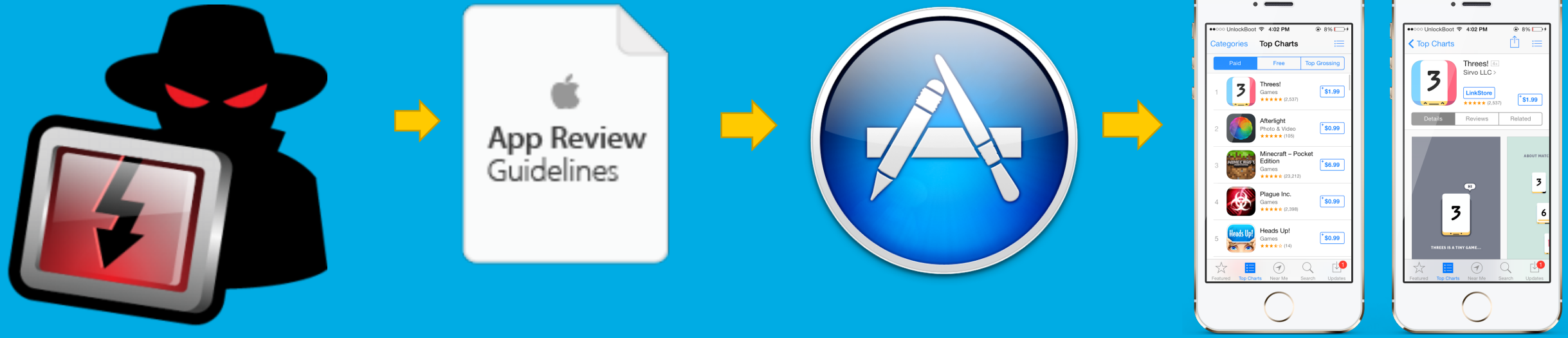
WHY IS IOS SO SECURE?

- Data execution prevention
- Encrypted file system
- Privilege isolation
- Sandboxing
- The main difference between iOS and android is
- **Mandatory app review**
- **Mandatory code signing**

RESTRICTED APP DISTRIBUTION

- All apps have to be reviewed by Apple.
- The apps that pass the review which searches for malicious activity and whether it violates apple agreements.
- All the apps that exist in App Store are checked and vetted by Apple.
- If the Apps do not have the sign then the app will not be run by the devices.
- The apps integrity cannot be changed after the vetting.
- iOS devices are only allowed to run apps downloaded through app store. (Unless Jailbroken)

FLOW OF VETTING



JEKYLL ON IOS

- The app is seemingly benign and was published on App Store.
- Jekyll can be instructed to carry out malicious tasks by reordering and rearranging the benign functionalities.
- The vetting is assumed to work by executing all the paths of execution by checking for malicious activity.
- So if we can change the control flow of the app then we can hide the malicious activity in plain sight.
- By this we know that the apple vetting though effective does not always identify the malicious apps

FORMS OF ATTACK THROUGH JEKYLL

Attack Type	Attack Description	Affected Version
Invoke Private APIs	Sending SMS	iOS 5.x
	Sending Email	iOS 5.x
	Posting Tweet	iOS 5.x & iOS 6.x
	Abusing Camera	iOS 5.x & iOS 6.x
	Dialing	iOS 5.x & iOS 6.x
	Manipulating Bluetooth	iOS 5.x & iOS 6.x
	Stealing Device Info	iOS 5.x & iOS 6.x
Attack Kernel	Rebooting system	iOS 5.x
Attack Other Apps	Crashing Mobile Safari	iOS 5.x & iOS 6.x

DRAWBACKS OF MALICIOUS APPS

- The drawbacks faced by apps like Jekyll and other malicious apps are:
- They do not garner enough user attention hence cannot infect large base of devices like other apps.
- These apps are mostly installed on accident and run on the same basis.
- If Apple is aware that such malicious apps exist, they could remove them from App Store immediately.
- They could also disable running of the app remotely through all devices.

MAIN WAY OF ATTACK

- The main way of attack that is discussed is infecting the iOS devices through infected window pcs by using botnets.
- We assume that the owner of the device is going to connect to the pc to sync, backup, restore data or upgrade firmware or just for charging.
- We assume the connection to be either through USB or by Wi-Fi based syncing.

USB or Wi-Fi based syncing

Syncing, backup, restore, upgrade

FAIRPLAY DRM

- Apple used DRM(Digital Rights Management) technology to prevent piracy of iOS apps.
- Three steps in running the iOS app are:

1. Verifying the apps code signature
2. Perform DRM validation and decrypt the executable file(Since all apps are encrypted by apple)
3. Run the decrypted code.

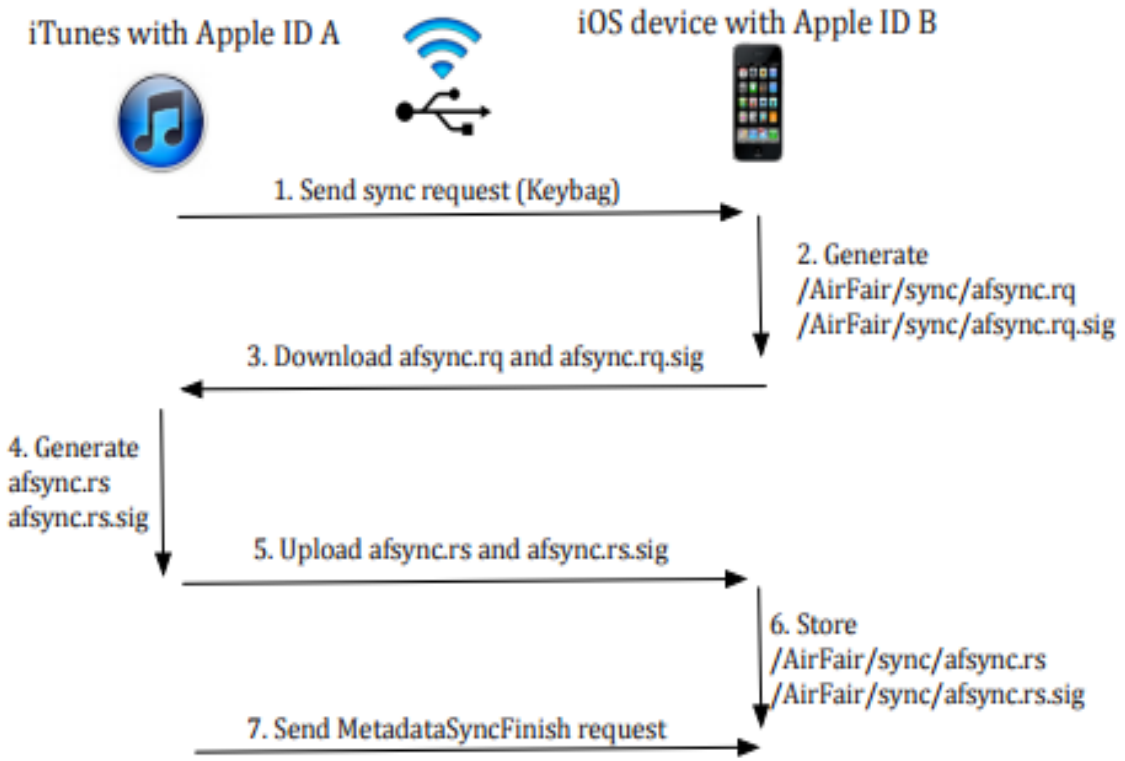
As a result copy of iOS app purchased by Apple IDa does not run on iOS devices of other Apple ID's.

FAIRPLAY DRM LOOPHOLES

- Different Apple IDs will receive the same encrypted executable files for different copies of the same app.
- iOS user will receive a file with the .ipa extension from the App Store. Although the whole ipa package is unique for each Apple ID, the encrypted executable files inside these ipa files are the same.
- This proves that the final decryption of the executables is irrelevant to Apple IDs of the device.
- It is also found that iTunes can sync apps in its app library to iOS devices through USB or Wi-Fi connections, even if the iOS devices are bound to different Apple IDs
- This means that when an iOS device with Apple IDb is connected to iTunes with Apple IDa, iTunes can still sync apps purchased by Apple IDa to the iOS device, and authorize the device to run the apps

FAIRPLAY DRM

1. Escrow keybag is used for iTunes syncing, This keybag allows iTunes to back up and sync without requiring the user to enter a passcode. When a passcode-locked device is first connected to iTunes, the user is prompted to enter a passcode. The device then creates an escrow keybag containing the same class keys used on the device, protected by a newly generated key. The escrow keybag and the key protecting it are split between the device and the host or server, with the data stored on the device in the Protected Until First User Authentication class. This is why the device passcode must be entered before the user backs up with iTunes for the first time after a reboot

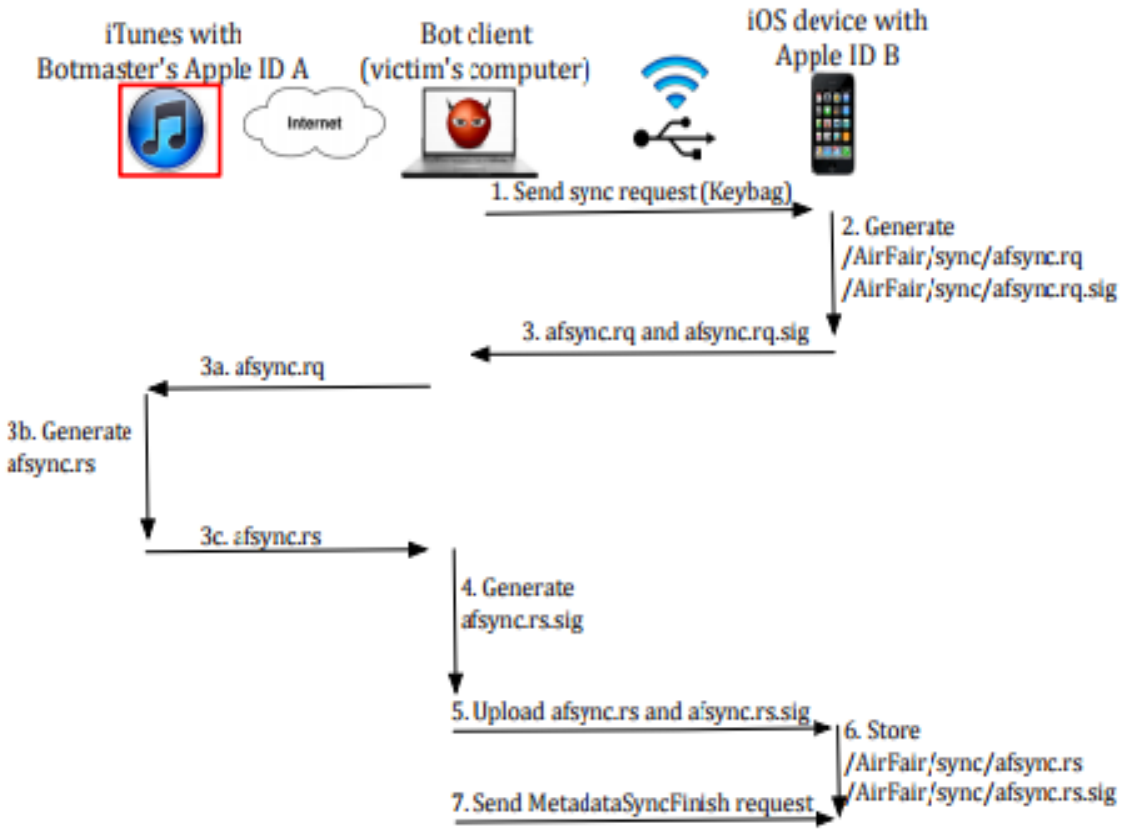


FAIRPLAY DRM

1. The iOS device generates an authorization request file /AirFair/sync/afsync.rq and corresponding signature file /AirFair/sync/afsync.rq.sig
2. Upon retrieving these two files from the iOS device , iTunes generates an authorization response file afsync.rs and corresponding signature file afsync.rs.sig .
3. iTunes then uploads the authorization response and signature files (afsync.rs and afsync.rs.sig) to the iOS device The iOS device stores the two files in the directory /AirFair/sync/ and updates its internal state.
4. Finally, iTunes sends a request to the iOS device to finish the syncing process.

MAN-IN-THE-MIDDLE ATTACK

- This working is same as the earlier but instead of the local pc producing the authorization file, it is sent to a remote pc which generates the authorization file afsync.rs and then send afsync.rs to the middle man.
- Hence the iOS device connected to a local computer obtains authorization to run apps purchased by the iTunes instance running on a remote computer.
- This technique is used to run the Jekyll app on different iOS devices with different Apple IDs without triggering DRM violation.
- The attack demonstrates that even if an app has been removed from the App Store, attackers can still distribute their own copies to iOS users.



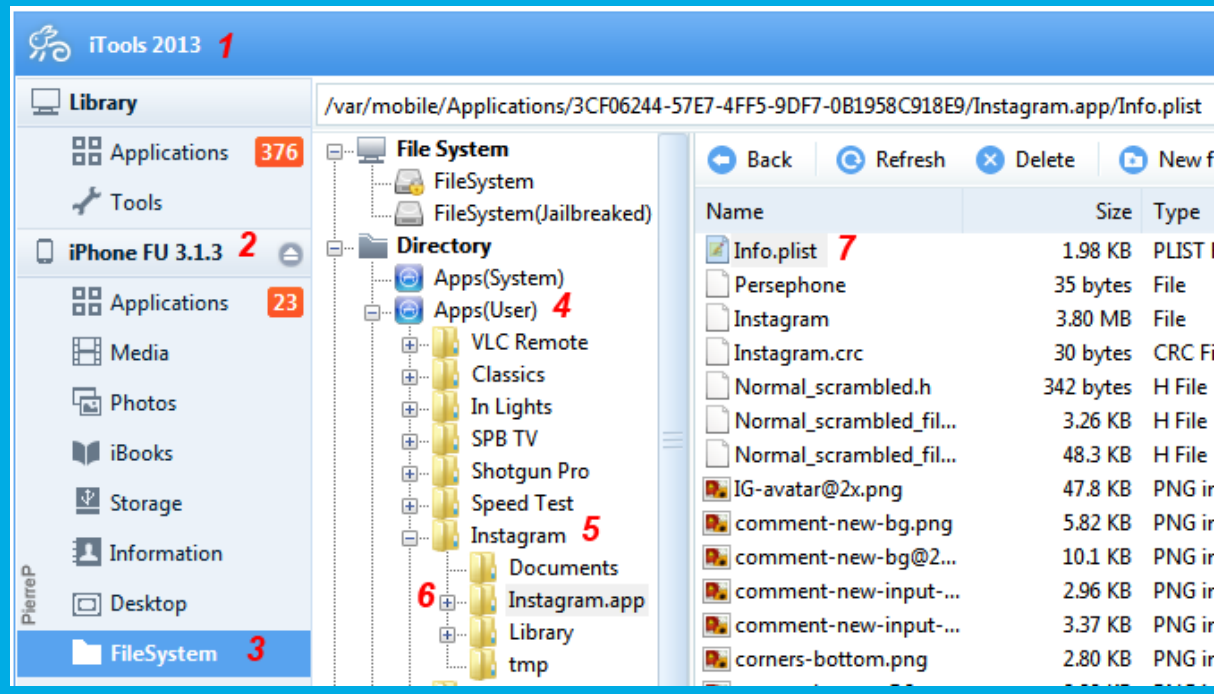
DELIVERY OF ATTACKER-SIGNED APPS

- Apple allows developers to install apps into iOS devices through a process called device provisioning, which delegates code signing to iOS developers.
- A provisioning profile is a digital certificate that establishes a chain of trust. It describes a list of iOS devices that are tied to an Apple ID, using the Unique Device Identifier (UDID) of each device
- However, we found that the installation of provisioning profiles can also be done by directly sending requests to a service running on iOS devices called "com.apple.misagent" launched via services like libimobiledevice or more tools.
- A compromised pc can be instructed to provision a plugged in iOS device without user knowledge.
- The removal of an app is done by issuing an Uninstall command and app-id to a service on the device called com.apple.mobile.installation proxy.
- Similarly installation of an app is done by issuing an Install command and app-id to a service on the device called com.apple.mobile.installation proxy

STEALING CREDENTIALS

- We know that iOS implements each app in a Sandbox environment.
- All the apps in the iOS devices have their own unique directories for their files and other apps are restricted to access it due to the restrictions of sandbox environment.

- Many apps like libimobiledevice or iTools use Apple File Connection(AFC) protocol to access data through USB cable.
- Many developers presume that iOS sandboxing is secure and store the cookies in plaintext which could be accessed by the attacker using tools mentioned(ex: Starbucks)
- The paper shows that by reusing the cookies, the attacker can log in as iOS user via web services for apps like Gmail and Facebook.



MEASUREMENT

- DNS Query Datasets:
- The data is collected from two large ISP's in US from 13 cities in 5 days.
- Client ID's(CID) that queried fewer than 1000 distinct valid domains are assumed to belong to home networks.
- If a CID queried any C&C domain in a day, we consider it as having a bot at home for that day.
- We utilized unique software update traffic to fingerprint Mac OS X and exclude them from the measurement.
- The iOS devices are identified when they access e Weather app, Stocks app, and Location Services.
- We found that because of the Apple Push Notification Service, iOS devices need to constantly query a certain domain name for push server configurations . We name this as e iOS heartbeat DNS queries.

MEASUREMENT

- DNS Query Datasets:
- To pinpoint Windows iTunes, our observation is that if we observe App Store purchases but do not find iOS heartbeat DNS queries, then the purchases must originate from iTunes. This identifies the Windows iTunes population.
- The Bot Population calculated for the day 10/12/2013 is 473,506 infected CIDs.
- Mac OS X CID is 6966(1.50%), so excluding this CID we have 466,540 bot CIDs.
- iOS CIDs are 142,907 which is 30.63% of the CIDs
- We further identified 112,233 CIDs with Windows iTunes purchases on the same day, so 112,233(23.70%) of CIDs have both iOS devices and Windows iTunes but no Mac OS X.
- This proves that 112,223 of CIDs are vulnerable to malicious attacks.

MEASUREMENT

- Of the 23% devices which are vulnerable there are bound to be devices which use banking applications.
- we chose mobile domains from eight banks :Citibank, Wells Fargo, PNC, Bank of America, SunTrust, Bank of the West, and U.S. Bank and examined how many of those iOS devices queried them. The result is that 4593(4%) of the devices accessed the banking domains.
- These devices which have existing banking apps could be replaced with malicious apps that look and feel the same way as the original ones to steal the user data and cause harm.

ACCURACY OF MEASUREMENT

- There might be more devices vulnerable cause this survey did not consider cellular traffic and that people can have multiple iOS devices in the same household.
- The data we have only allows us to determine what type of devices are behind an IP address, but not how many of each. So there is a possibility that there may be multiple Windows machines and that not all of them maybe infected.
- Due to the mobility of iOS devices, it is possible that the same iOS device appears in different “infected” IP addresses, which leads to an overestimation of the number of potential iOS victims.

PREVENTION

- Due to the sheer number of apps in App Store and lack of run time monitors on iOS devices, malicious apps are only detected when the user detects them.
- Apple should monitor the anomalous Apple IDs that deliver purchased apps to excessive number of devices and verify them.
- The iOS should also warn the user when app purchased by different Apple ID is being installed and let the user authorize it.
- The iOS should warn the user when a provisioning profile is installed or prompt the user the first time an app is run that is signed by an unknown provisioning profile.
- Third-party developers should be aware that plaintext credentials/cookies could be easily leaked through the USB interface and store the credentials in a secure manner to prevent leaks.

CONCLUSION

- This paper discussed the feasibility of large scale infection of iOS devices.
- It shows that even the Apple signed apps can be malicious and shoes that iOS is not as secure as it seems and there are ways around it.
- It also demonstrates different kinds of attacks against the devices using a compromised computer: delivering Apple signed malicious apps, delivering third party developer signed malicious apps and stealing of private data and credentials from iOS devices.
- It also shows that 23% of the CIDs could possible be infected through compromised systems.

THANK YOU

On the Feasibility of Large-Scale Infections of iOS Devices.

Tielei Wang, Yeongjin Jang, Yizheng Chen, Pak-Ho Chung, Billy Lau, and Wenke Lee. In UsenixSecurity'14.

Paper Discussion

- Zhenyu Ning
- CSC 6991 – Advanced Computer System Security
- Usually, we consider iOS system as a much safer system comparing with Android system by the contribution of both the strict audit and sandbox protection on applications. But in this paper, author break the conventional view by representing an approach to install arbitrary malicious applications to the victim's device which is connected to a compromised computer and steal sensitive data from the device.
- Generally, this achievement is accomplished by a man-in-the-middle attack. Once the iOS device is connected to the compromised computer, the attack forces the iTunes to send a sync request to the device. After the computer get the response of the request, attack then force the computer to send the response to the attack's computer instead of traditional action which will send an upload request to the device. The attack then sends the upload request from his computer to the iOS device through the compromised computer to accomplish the authorization. Once the authorization is setup remotely, the attack then can delivery arbitrary applications to the device without the sense of the victim through provisioning process which is original designed for developers to debug their applications. Also sensitive data like credentials in the cookies can be stolen in the same way.
- Also the author gives a measurement about how many devices may suffer from this kind of attack by analysis of records of DNS request and network traffics. The result shows that about 23% of bots are perform this attack to the iOS devices connected to it.

Paper Discussion

- Sai Tej Kancharla
- CSC 6991 – Advanced Computer System Security
- The paper "On the Feasibility of Large-Scale Infections of iOS Devices" by Tielei Wang, Yeongjin Jang, Yizheng Chen, Simon Chung, Billy Lau, and Wenke Lee discusses briefly about the security in current iOS and how it is vulnerable to malicious apps and also shows that it is feasible to large scale infections remotely using botnet.
- The paper discusses about the ways in which Apple signs and ensures the integrity of the app in the App Store. It shows the design flaws in the vetting mechanism which enables the attacker to submit malicious apps to App Store. The general idea of Apples verification of app is to check whether all the existing paths contain any malicious data. The paper shows even after removal of the malicious apps from App Store the user can spread the app through third party developer signed certificates. It also exposes the loopholes in the Fairplay DRM protocol which is used to deliver malicious apps remotely to the iOS device and also steal data.
- The loophole found is that iTunes can sync apps in its library to iOS devices through USB or Wifi Syncing even if the devices are bound to different Apple IDs. The Man In The Middle Attack works using this principle to deliver malicious apps. Once the iOS device is connected to the compromised computer, the device generates an authorization file and corresponding signature file. Upon receiving the files the victims computer sends the response to the attackers computer remotely for the authorization response file and its corresponding signature file . The remote attackers pc sends the authorization response which is uploaded to the device by the victims pc. This enables the user to install the malicious apps without causing any DRM violation. The attacker can also install provisional profile into the device remotely and this enables the attacker to install/uninstall apps remotely. The attacker can replace the apps with malicious apps which look and feel the same to steal the users data.
- The paper also calculates the number of devices that can be affected by connecting to compromised computers by analyzing the DNS query datasets of two large ISPs and calculating the number of iOS devices which are connected to the compromised Windows iTunes. The analysis shows that out of the 473,506 infected CIDs 112,223 CIDs which is 23% devices are vulnerable to attack. The paper also presents on how the Apple can improve the analysis of the apps and how to control the Man In The Middle attacks to protect sensitive data

Paper Discussion

- Sharani Sankaran
- CSC 6991 Advanced Computer Security
- This paper mainly describes that the iOS Apple has increase attention from attackers due to its popularity and the device provisioning process and in-file storage was mainly installed so that a compromised computer may install an apple signed malicious app on a connected iOS device.
- In this paper It mainly challenges the common belief that the Apple App Store is the sole distributor of iOS apps. Instead of relying on tricking a user into downloading apps from the App Store, attackers can now push copies of their app onto a victim's device.
- Second, this expolit challenges the common belief that the installation of iOS apps must be approved by the user. Attackers can surreptitiously install any app they downloaded onto victim's device
- we mainly analyze DNS queries generated from more than half a million anonymized IP addresses in known botnets.
- This paper also determines measurement about how many devices may suffer from this kind of attack by analysis of records of DNS request and network traffic.by analyzing DNS queries generated from more than half a million IP addresses in known botnets, we measured that on average, 23% of bots are likely to have USB connections to iOS devices, potentially leading to a large scale infection.

Paper Discussion

- Lucas Copi
- CSC 6991
- 19 October 2015
- iOS Security
- The paper *On the Feasibility of Large-Scale Infections of iOS Devices* discusses the ability of attackers to bypass Apple's app authorization methods and infect iOS devices through wireless sync or through a USB connection to iTunes. The researchers discovered a vulnerability in the iTunes syncing mechanism that allows attackers to bypass DMR checks and carry out man in the middle attacks. Researchers also found a vulnerability allowing iOS devices to be provisioned for development through USB connections. This allows attackers to replace legitimate apps with malicious third-party apps.
- The paper demonstrates the capability of wide spread attacks against iOS devices by collecting data about the number of iTunes purchases on Window's machines and then making assumption about the number of iOS devices tied to these accounts. Once the paper demonstrates the feasibility of large scale attacks, it goes into detail on the methods used to carry out the three main attacks in the paper: apple-signed malicious attacks, delivering third-party malicious apps, and stealing private user data through these vulnerabilities.

Paper Discussion

- Hitakshi Annayya
- Paper Summary of On the Feasibility of Large-Scale Infections of iOS Devices Tielei Wang, Yeongjin Jang, Yizheng Chen, Simon Chung, Billy Lau, and Wenke Lee, Georgia Institute of Technology
- Because of the advanced iOS's security architecture, there are many flaws in design, iTunes syncing process is slow etc. When a compromised computer is connected to iOS device, simply we can install Apple-signed malicious apps and attack and steal data from Facebook, Gmail apps cookies. After analyzing from DNS queries, we got to know 23% of bot IP address has connected to iOS devices and thus making large scale infection feasible.
- Despite the advanced techniques i.e powerful revocation capabilities, mandatory code signing mechanism, the Digital Rights Management (DRM) technology which are integrated by iOS devices, infecting a large number of non-jailbroken iOS devices through botnets is feasible.
- The main contributions of author's work are discover a design flaw in the iTunes syncing process, and present a Man-in-the-Middle attack that enables attackers to run any app downloaded by their Apple ID on iOS devices, second, the security implications of the stealthy provisioning process and insecure credential storage and finally a large scale infection of iOS devices is a realistic threat and we are the first to show quantitative measurement results.
- Later the author's discusses on the methodology and datasets we use to determine a lower bound of the coexistence of iOS devices, App Store purchases made from Windows iTunes, and compromised Windows machines in home networks, with a goal to quantitatively show that a large number of users are likely to connect iOS devices to infected personal computers. Finally, 23% of bots are likely to have USB connections to iOS devices, potentially leading to a large scale infection.

iOS papers in CCS'15

- **Cracking App Isolation on Apple: Unauthorized Cross-App Resource Access on MAC OS X and iOS**
- Luyi Xing (Indiana Univ. Bloomington); Xiaolong Bai (Indiana Univ. Bloomington & Tsinghua Univ.); Tongxin Li (Peking Univ.); XiaoFeng Wang (Indiana Univ. Bloomington); Kai Chen (Indiana Univ. Bloomington & Chinese Academy of Sciences); Xiaojing Liao (Georgia Institute of Technology); Shi-Min Hu (Tsinghua Univ.); Xinhui Han (Peking Univ.)
- **iRiS: Vetting Private API Abuse in iOS Applications**
- Zhui Deng (Purdue Univ.); Brendan Saltaformaggio (Purdue Univ.); Xiangyu Zhang (Purdue Univ.); Dongyan Xu (Purdue Univ.)

Reminders

- Proposal Revision Due on Wednesday, Oct. 21.
- Paper Summaries