Security Implications of iOS

Kiel Wadner

Security Implications of iOS

GIAC (GSEC) Gold Certification

Author: Kiel Wadner, kiel@timeisswift.com

Advisor: Tim Proffitt

Accepted: July 27, 2011

Version 1.1

Abstract

Starting with the iPhone's release in 2007, smartphones have greatly increased in popularity and are approaching 25% of the mobile market. The increased popularity is forcing more organizations to deal with the security of their data on both personal and organization owned phones and tablets. It is therefore, critical the implications of smartphones in relation to controlling an organization's data be understood. This paper looks at several threats against iOS devices, the ways they can be mitigated, and what it means for an organization's data.

Kiel Wadner, kiel@timeisswift.com

1. Setting the Stage

Security is mostly a superstition. It does not exist in nature, nor do the children of men as a whole experience it. – Helen Keller

1.1. A Prime Target

In the last two years, smartphone market penetration has gone from 16% to 23% (Kellog, 2010). Gartner is predicting that, "Worldwide mobile voice and data revenue will exceed one trillion dollars a year by 2014" (Gartner, 2010) with smartphones leading sales in mature markets. This growth paired with increased functionality and access to data will make smartphones a prime target to attackers.

1.2. Implications

An implication is, "*the conclusion that can be drawn from something, although it is not explicitly stated.*" or, "*a likely consequence of something*" (New Oxford Dictionary, 2010). The goal of this paper then is to discover what effect iOS devices will have on an organization's data - what the likely consequence will be. This is a tall order given the diversity between organizations, their data, and their policies.

The saying goes, "It's not a matter of if a system will be compromised, but when." As security professionals, the job is not to simply prevent systems from being compromised, but rather to greatly reduce the odds of it happening and the impact when it does. With that in mind the key implication is – will iOS devices make our data more vulnerable than other methods of access? The answer is not purely technical. Users, procedures and even organizational culture will affect the overall security of an environment.

1.2.1. Disclaimer

Mobile security and the iOS operating system are large topics. The focus of this paper is on their security implications as a whole. If you are interested in deployment guides Apple has provided a very good iPhone Deployment Guide on their website.

KielThomas Wadner, kiel@timeisswift.com

2. What Are we Afraid of?

The security concerns for an iOS device are, for the most part, familiar. They are the same risks we'd expect on any end-user system. In a recent report, ENISA presented ten risks common to mobile devices (European Network and Information Security Agency, 2010). Of those, I identified four threats that most directly impact the confidentiality or integrity of the data a device has access to:

- Software with a malicious intent
- Forensic analysis after theft or loss of a device
- An attacker stealing or modifying data in transit
- The user themselves

As stated, these are not earth shattering, or unexpected. In the introduction to *Mobile Malware Attacks and Defense* Ken Dunham (2009, p.2) correctly states that, "Since at least 2000 select security experts have predicted gloom and doom about pending future attacks against smartphones and other mobile devices. In large part they were wrong […]". It is correct that in the last ten years large scale issues haven't captured media attention. Still, the growing popularity of mobile devices is causing those warnings to be intensified and felt by businesses. In a 2010 study, eight out of ten CIO's stated they believe "smartphones in the workplace increase a business's vulnerability to attack, and rank data breaches as a top security related concern" (Schwartz, 2010). Dunham himself later states the problem is likely to get worse.

2.1. Software With a Malicious Intent

Malware is a serious problem to information security. It is a key way attackers gain access to a system and wreck havoc. A study by the Ponemom Institute and Panda Security in 2010 placed the cost of malware at $3.8 million dollars a year (Santana, 2010). The good news is this problem has yet to be realized on Apple's iOS devices. Mikko Hypponen, Chief Research Officer of F-Secure

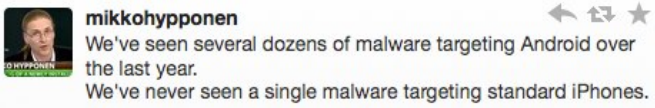

Figure 1

made a point of this in a June 1, 2011 Twitter post prior to a conference where he spoke on mobile malware.

There are a several factors that can contribute to the lack of malware on iOS devices. The first is the lack of propagation methods. The primary methods for malware distribution on mobile devices have been through Bluetooth, e-mail, SMS, and device synchronization (Dunham, et al., 2009, p. 6). As example, an early piece of malware for the Symbian OS, Cabir, spread over Bluetooth, but then asked permission from the user to install it. Learning from the past, Apple does not allow devices to transfer files over Bluetooth.

The App Store also limits another form of propagation, as it is sole way to install programs on a non-development or jail broken device. The controlled nature of the App Store raises the bar necessary to spread malware. Every application in the store is signed with a cryptographic signature tied to the author and must also be signed by Apple. In theory, this gives a paper trail to the original author, and if they are selling the application, a financial one as well.

There are still vulnerabilities that can be exploited. In the 2011 Pwn2Own contest Charlie Miller, a security researcher with *ISE*, was able to compromise an iPhone running iOS 4.2.1 through a vulnerability in WebKit (Naraine, 2010). WebKit is the open-source toolkit used in Safari and has been a frequent source of vulnerabilities. The iOS 4.2 release included 42 fixes (Apple, 2011), and 4.3 had 49 (Apple, 2010). Since Safari is a primary way users interact with data, it is safe to expect it will continue to be an area of interest to attackers.

Jail breaking a device is not covered in depth, but it is important to understand the process is possible because of vulnerabilities in iOS. Such vulnerabilities could be used for malicious intent. Also, the jail breaking process involves loading a modified kernel onto the device, which itself could compromise the data. There is no way to prevent jail breaking; the best that can be done is to audit devices manually or through management software.

2.1.1. Not All Malicious is Apparent

It is important not to pigeonhole "malware" as virus/worm/trojan, etc. Other classifications of software can access our data on purpose or by accident. In his Blackhat presentation Nicolas Seriot (2010) outlines several privacy concerns with iPhones. They were cases of how any application could access more private information then they actually need or should have. Examples of information include recent Safari searches, address book contacts, and YouTube history. Many applications have legitimate reasons to access that data so it is unlikely Apple will prevent it in the future. It is to be expected that spyware type applications will pop-up now and then, getting past Apple's review process, and might get access to private information for a time. Some organizations will feel more comfortable by creating a whitelist of allowed applications and restricting who can install them.

2.2. Forensic Analysis

There is a wealth of information on iOS devices, everything from the obvious call history, emails, contacts, and documents to the more obscure access point information, passwords, and GPS history. Your iOS device is not an open book – there are ways to secure it and they are discussed later. It is still important to understand what is on it should a compromise occur. The implications to security are based on what data exists and how hard it is to get it.

A large majority of the useful information is stored as either a .plist file or as an SQLite database. Apple's *Property List Editor*, which comes with XCode will open the plist files, and the databases can be open by the command line *SQLite* tool or a program like *Base* from Menial Software.

2.2.1. Some Interesting Items

A quality listing of what can be found is not possible for the scope of this paper. Some interesting items are below, but if you want more information I'll direct you to two books: *iOS Forensic Analysis* by Sean Morrissey and *iPhone Forensics* by Jonathan Zdiarski. Both are great resources and were used extensively while researching.

Map History: The history of map and direction searches is found at /mobile/Library/Maps/History.plist. Figure 2 is snap-shot of what was stored when I searched for the "Old Spaghetti Factory" in Portland OR. Note the GPS coordinates in case you are in the mood for quality food.

▼Item 1	Dictionary	(11 items)
DisplayQuery	String	Old spaghetti factory
HasMultipleLocations	Boolean	☑
HistoryItemType	Number	0
Latitude	Number	44.87141
LatitudeSpan	Number	2740631
Location	String	Portland
Longitude	Number	-122.9974
LongitudeSpan	Number	5625000
Query	String	Old spaghetti factory
SearchKind	Number	2
ZoomLevel	Number	9

Figure 2

Known Networks: A list of known network connections can be found under /mobile/SystemConfiguration/com.apple.wifi.plist. An example entry is shown below. This information can be useful to an attacker in finding out what connections are available in a network as well as the credentials to connect (those are stored in the Keychain)

▼List of known networks	Array	(8 items)
▼Item 0	Dictionary	(24 items)
AGE	Number	0
AP_MODE	Number	2
ASSOC_FLAGS	Number	1
BEACON_INT	Number	10
BSSID	String	0:40:[redacted]
CAPABILITIES	Number	1057
CaptiveNetwork	Boolean	☐
CHANNEL	Number	6
CHANNEL_FLAGS	Number	8
enabled	Boolean	☑
lastAutoJoined	Date	Mar 1, 2011 9:33:03 AM
lastJoined	Date	Aug 2, 2010 7:33:30 PM
NETWORK_HIDDEN	Boolean	☐
▶networkChannelListKey	Dictionary	(1 item)
NOISE	Number	0
▶RATES	Array	(12 items)
RSSI	Number	-72
ScaledRate	Number	1
ScaledRSSI	Number	0.6254870891571045
SSID	Data	<48656d65 [redacted] 26b>
SSID_STR	String	[redacted]
Strength	Number	0.6254870891571045
WiFiNetworkIsSecure	Boolean	☐
WiFiNetworkRequiresPassword	Boolean	☐

Figure 3

Contacts: Contact information is stored in several tables in the /Library/AddressBook/AddressBook.sqlite database. The *ABPerson* table contains a listing of people as seen in the figure. Phone numbers are stored in the *ABMultiValue* table.

```
1 select ROWID, First, Last, Organization, CreationDate from ABPerson
```

2 queries completed.

ROWID	First	Last	Organization	CreationDate
1	Bob	Smith	Acme	330565449
2	Jane	Doe	NULL	330565480

Figure 4

KielThomas Wadn

2.3. An Attacker Stealing or Modifying Data in Transit

Many attacks rely on weaknesses in transit. An armored truck is less secure than a bank vault, military hardware is more at risk in transport then on a base, and data roaming the Internet will see more threats than on a backup tape on a shelf. The portability of iOS devices makes them an ideal target for man-in-the-middle attacks, or general traffic sniffing. As the user travels around they will be using a variety of locations to get access. They are also more likely to use the devices at open Wi-Fi connections, which will not have the same security requirements as your internal network. For wireless access the iPhone 4 has an 802.11n antenna and the iPhone 3GS comes with a b/g antenna (iphone wiki, 2011). For both devices iOS will prefer Wi-Fi to a data plan if enabled.

Organizations tend to send and receive email over TLS and access internal web pages over https, but many (particularly smaller ones) don't provide secure instant messaging. This leaves users on 3rd party services that are not secured. While not ideal on your internal network, the problem is huge if they are using an IM tool on their phone while on the airport or coffee shop wireless network. Understanding what tools your users will utilize when accessing data will allow you to decide what steps are needed to secure it.

2.4. The Human Risk

Depending on the environment, it is likely that users will be one of the greatest risks. This is because most systems require interaction and no matter how many controls are put in place, a level of trust and power is given to the user. This remains true for iOS devices. There are three areas that are key to the human targeted attacks: social engineering, lack of understanding, and apathy.

2.4.1. Social Engineering

End users are becoming more aware of phishing emails, fake websites and other social attacks when they are on their computer. However, this may not be true for mobile devices. Researchers are realizing that the visual cues users expect are not present on

mobile devices, which may cause them to miss a social engineering attempt. The report out of UC Berkeley concluded that, "[…] of 100 mobile applications and 85 web sites finds that mobile applications and web sites commonly interact in ways that can be spoofed by attackers" (Porter Felt & Wagner, 2011). This makes it hard for the user to understand what is a fake and what is not. It also makes the job of IT harder when training the user what to be on the lookout for.

2.4.2. Lack of Understanding

This problem is similar saying most drivers don't understand their cars as well as mechanics. It is not an insult to the driver, just a fact since they don't work on cars day-in and day-out. It is not expected that they will understand it to the same level. Some understanding is needed so they don't damage their cars to excess, but few people need to understand the mechanics behind a transmission.

Similarly, most users of smartphones don't understand the security details like someone whose job is in information security. The lack of understanding can cause a user to make mistakes that have adverse implications on organizational data. This may be a result of training or exposure. As an example, most users won't think about what private data a 3rd party application might get access to or how easy a 4-digit passcode would be to brute force. It is not their fault per say, but something necessary to keep in mind when training so the correct level of understanding is obtained.

2.4.3. Apathy

People have busy lives, and they want tools and technology that help them get their jobs done. They don't really care about the level of encryption used, or how long it will take to brute force a passcode. They don't view the risks in the same light as someone in InfoSec. For iOS devices, their simplicity could cause an increase in apathy about security on them.

Like a lack of understanding, a certain level of apathy is expected and natural. It is likely that someone who deals with, and sees the side effects of diabetes on a daily basis is going to care about it more than others. Those who don't will be more apathetic

about it. The apathy must be overcome when providing training, or a defense is made for the decisions and policies in place.

3. The Defenses

3.1. The Human Solution

We ended looking at the human threat, and it is no coincidence that we start with the human solution. Some readers may disagree but training the user is the single most important step for data security. They are the ones interacting with the technology and controlling the data, and the best security in the world will fail without their involvement. The word training was purposefully used over the more common "security awareness program" phrase. This is to emphasize simply being aware of the problem isn't going to make it go away. A parallel is the growing diabetes problem in America. Health professions can spend a lot of time and money making people aware of the problem, but if they don't persuade the individual to take action themselves and train them in better health practices diabetes will get worse.

When building your training material be sure to spend time talking about mobile phones – even if they are not formally supported. Cover the topics mentioned in here such as passcode length, notifying immediately if the device is lost, setting it up for remote wipe, and other key mobile security topics. You can talk about mobile devices when covering other topics such as email security, using Wi-Fi hotspots and keeping your system up to date.

3.2. Encryption

Apple first introduced hardware encryption with iOS 3 (Morrissey, 2010, p. 29) however it was found to have significant flaws. Jonathon Zdziarski showed in his book and in YouTube videos, how the encryption could be by-passed by jailbreaking 3G[S] devices (Zdziarski, 2009). With the introduction of iOS 4 Apple resolved the issues and until recently it was generally believed a secure option. On May 23, Vladimir Katalov, CEO of ElcomSoft, announced they had found a way to decrypt the hardware protection

(Katalov, 2011). This is significant because it revealed the data was not as secure was many thought.

It's important to point out that the encryption used wasn't broken; what Elcomsoft was able to do is extract the keys and brute force the passcode. To understand the state of encryption given that, let's take a step back and look at what encryption is provided by iOS. This is a bit difficult since there aren't many public sources of information not covered by Apple's confidentiality agreement.

3.2.1. Hardware Encryption

Hardware encryption in iOS 4 is implemented with 256-bit AES encryption (Apple, 2010) and is always enabled. The file structure itself is not encrypted but each file is encrypted with its own unique key which is derived from the device itself (Vance, 2010). By setting a passcode, what Apple calls "Data Protection" is enabled. This allows files to be encrypted with both the device keys and one derived from the passcode (Katalov, 2011). The benefit being files are encrypted with something you have, the device, as well as something you know – the passcode. This is a good security measure on Apple's part and acknowledges the weakness of only using keys stored on the device. The catch is it is up to the developer to employ the encryption APIs for this to take effect, and for the user to choose a reasonably complex passcode. The second part is difficult since iOS defaults to a 4-digit passcode. This should be changed to allow/ require more complex passcodes. Currently the Mail application is the only iOS 4, built-in application which uses Data Protection.

According to Katalov, what they were able to obtain is the device-based key (either by finding it or computing it) and then brute forced the user's passcode on the device itself. Once they have those key's they are able to decrypt the data. In email exchange with Katalov, he confirmed this does include Keychain information. It is the keychain that is used to store private information such as VPN passwords, email credentials, and anything else an application chooses to store there. This is an obvious security problem, and although ElcomSoft is limiting who has access to the device it can be expected other organizations will be able to duplicate the results.

There is a non-technical way to gain access to an encrypted device – ask Apple. If Apple is given a device and warrant they will remove the passcode lock, allowing full access (Morrissey, 2010, p. 87). This isn't useful to an individual but interesting for several reasons. First, in the case of an insider threat, if legal action is taken against them the device data can be pulled and used in court even if they don't wish to turn them over. Second, and perhaps most obvious is Apple removes the passcode! It's not known (at least I couldn't find) how they go about this. That does imply it is at least *theoretical* that someone else could do this. Given what ElcomSoft was able to accomplish this might be a moot point, but it is still another avenue of attacking the encryption.

3.2.2. iTunes Backups

A large amount of information is accessible from the backups created by iTunes, which are not encrypted by default. The reasoning being if an attacker has physical access to the sync'ing computer, they already have access to the data and probably more. This makes sense, but not entirely true. Several private pieces of data typically exist on a device but not a computer. Examples are: phone call log, random (maybe private) pictures, location information, and saved credentials. The last one is interesting. Even security aware people may let their phone remember passwords that are long and complicated. For those reasons, it is recommend that iTunes back-ups be encrypted. This can be done under *Options* on the main iOS device page in iTunes.

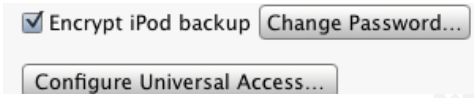

Figure 5

If the data is important enough to encrypt, it is important enough to use a strong and complex password. ElcomSoft has another tool, its *Phone Password Breaker* that uses a combination of dictionary and brute force attacks against iOS device back-ups (ElcomSoft, n.d.). At rates of 35,000 passwords per second simple passwords will not last long. Further, the professional version can decrypt the Keychain data store where passwords and other "secure" items are usually placed. Imagine a piece of targeted malware where the target is the back-up file. Paired with some decryption tools they could get access to account information, VPN logins, and a whole variety of things.

3.2.3. Remote and Local Wipe

All devices should be set to perform a secure wipe after a number of failed login attempts. Four or five attempts will be sufficient in most cases.[1] This is much lower than Apple's default 10, but after the 6th and 7th attempt one has to wonder if the user had already forgotten the passcode and it no longer matters. For your user's sake inform them of the policy up-front and explain why it is important to frequently sync their device. A secure wipe can also be performed remotely should the employee lose the device, or is terminated.

For iPhone 3G the secure wipe performs a traditional overwrite of the user data partition by setting the bits to 1's, a process that can take a couple hours. 3GS and 4G devices (with iOS 3 or later) has the hardware encrytion and simply overwrites the encryption key. This occurs quite quickly. With the key removed, the encrypted data cannot be accessed, but remains on the device. It might as well be random data without meaning. In talking to Katalov, he confirmed that Apple's process for this was secure. That doesn't mean added security can't be taken and over-write the data on the device. In an email exchange with Amber Schroader of Paraben Corporation, she said her opinion is to perform such a wipe. "We do a full overwrite of the data on the phone for the most secure option. Then you can re-flash the device with the OS. This is the best method to make sure that all potential user data is removed from the device." This does make sense if the device is sold, or reused to be certain the data is gone. Mounting an iPhone as a drive is not straight forward. Amber was understandably not able to revel how this is done at Paraben, but there are at least two ways to accomplish it.

First is to use a tool like iPhoneDisk (Porter, 2010) which was written by Allen Porter, a Google employee. It allows you to mount the iPhone user partition as a folder and then access it through Terminal. From there, '*srm'* or similar tool can erase files or folders from the user portion. iPhoneDisk relies on MacFuse which is a SDK for using 3rd party file systems on OS X (10.4 and later). Perhaps because it lives in "user space",

[1] This does make the device more susceptible to the "3-year old child" attack, and back-ups should be regular

it does not show up as an actual device disk. Unfortunately this means using disk level tools is not an option. If it were, tools such as '*dd'* could be used to write random data to the entire drive. The second option is to jailbreak the device if an exploit method exists for your current device and iOS version. The cat-and-mouse game continues on that front and changes rapidly.

3.3. iOS Protecting Itself

After the initial release of iOS, Apple made significant strides securing how the operating system controls code access. These actions have made it harder to execute an exploit on a device.

3.3.1. Sandbox

A software sandbox is a mechanism to restrict the access that a program has. Apple's sandbox technology (for OS X as well as iOS) is based on TrustedBSD project (Blazaki, 2011) and is made up of user-land functions to setup the sandbox. According to Apple (2010):

> "The sandbox is a set of fine-grained controls limiting an application's access to files, preferences, network resources, hardware, and so on. Each application has access to the contents of its own sandbox but cannot access other applications' sandboxes. When an application is first installed on a device, the system creates the application's home directory, sets up some key subdirectories, and sets up the security privileges for the sandbox".

It's key to understand the sandbox doesn't limit access to core application data such as Safari, or YouTube (Seriot, 2010). This would allow third party applications access to that information and doesn't prevent the spyware problem.

3.3.2. Data Execution Protection

DEP is a common technique or preventing malicious code from running. It has been used in XP since Service Pack 2 (Microsoft, 2006), and has existed in iOS since version 2.0 (Miller, n.d.). In Miller's words:

> "The way that DEP works is that it allows the processor to differentiate between code (the stuff that 'runs') and data (stuff like pictures, words, etc. that is not supposed to 'run'). This makes it hard on an attacker because they'd like to inject some code into an application disguised as data and then by using a vulnerability get that code to run."

It's not a foolproof solution as Miller showed in the 2011 Pwn2Own competition where he by-passed DEP in a 4.2.1 device to exploit it. (Naraine, 2011). In that case, the device was compromised by visiting an website in Safari. Miller noted that the exploit would have failed against 4.3 devices (which released around the time of Pwn2Own 2011) because it implemented partial ASLR.

3.3.3. Address Space Layout Randomization

Tino Muller has a great summary of ASLR in his paper, "ASLR Smack & Laugh Reference" (Muller, 2008). He says:

> "How does address space layout randomization work? Common exploitation techniques overwrite return addresses by hard coded pointers to malicious code. With ASLR the predictability of memory addresses is reduced by randomizing the address space layout for each instantiation of a program. So the job of ASLR is to prevent exploits by moving process entry points to random locations, because this decreases the probability that an individual exploit will succeed."

ASLR was added in iOS 4.3 and has greatly raised the bar required to both jailbreak and maliciously exploit the device. I expect that we will see further advancements on the ASLR front on iOS devices due to its track record of making it hard to run malicious code on a variety of systems.

3.4. Custom Application

Depending on the size of your organization and the type of data to be secured a good option is to develop a custom application. It can be designed to expose a subset of your data, only cache certain parts, and make sure that Data Protection is used for any stored data. If your software architecture already exposes a web service, then that can be

leverage to access the data. This will decrease the spin-up time needed for development. Ideally, the web service is accessible by internal IPs only, and you can take advantage of VPN options that are covered later.

Through the iOS Developer Enterprise Program (Apple, 2011) your organization can deploy applications internally to your own workforce and by-pass the iTunes store. If you will only need the application on 100 or less devices you can use the ad-hoc deployment option with a standard developer account.

3.4.1. Advantages

The primary advantage of a custom application comes from the ability to have complete control of the life cycle of the data on the device. This starts with validating only a secure connection is used to get data, and ends with securely storing, or wiping the data. When web sites are viewed through Safari, a cache and history is kept, potentially storing parts of the data on the device. There exists the possibility that this cache could be viewed by malware, or by an attacker with physical access to the device. Custom applications can either cleanup after itself, or use the file-level encryption to further protect the data.

Further, by providing a secure storage of the data, functionality can be built to take advantage of offline data. Even in today's world there are times when a reliable data link is not available. If many of the devices are personally owned by employees, and the enterprise has limited control over them, a custom application provides a way to control access to the data in a pseudo-sandbox. With a personal device, the organization may not have the luxury of wiping the phone if lost – assuming they even hear about it. In this case, a custom application allows some degree of certainty business data is still secure.

A further measure that compliments a custom application is to remove Safari. This may seem drastic, but if the device is owned by the company, used for specific purposes and the tools are provided through a custom application, this might be a reasonable action. In removing Safari, a major vector of threats is also removed. This is seen by the number of WebKit related vulnerabilities patched in the 4.2 release.

Removing Safari also removes key way rouge employees could leak information - though web based email or file sharing.

A custom application is not a silver bullet, but it is worth considering if there is a subset of business data that should be viewed or interacted with.

3.4.2. Disadvantages

Building a custom application does require developers with skills in Objective-C and familiarity with the iOS SDK. For many organizations this is not available in house, requiring existing employees to be training or the project contracted out. That cost may not justify the gains in some organizations. Software development is also a complex task, and to develop a secure application is not trivial. Bugs will be found and need updated, a detailed code audit will be required and a long term commitment from management and the development team will be required for success.

Further, many environments don't have the luxury of consolidating to a single platform. This could mean an iPhone application will not meet the business needs if Android, Blackberry, etc. phones are in common use. In that case, building a featured application for multiple platforms will not be feasible except in the largest of environments.

3.5. VPN

For organizations that need their users to access local resources, a VPN is the best way to accomplish this. VPNs have been used with laptops and other mobile devices for many years and lend themselves well to being used on a smartphone. From Apple's documentation, "Secure access to private networks is supported on iOS 4 using Cisco IPSec, L2TP over IPSec, Juniper, F5, and Cisco SSL VPN, and PPTP virtual private network protocols" (Apple, 2010) This means that most businesses that have a VPN concentrator deployed will be able to leverage it. By doing so, employees can securely access internal web applications or web services over both public WiFi, or using their data plan.

When dealing with public Wi-Fi, a man-in-the-middle attack is a real concern. Given that users historically don't do a good job of verifying certificates or other visual clues that they are on a secure connection, the VPN tunnel can assure their traffic is not intercepted in transit. A VPN connection will also prevent a mixed-mode authentication attack such as using SslStrip which would be more likely on public network.

A nice feature that Apple provides on their devices is VPN on Demand. In a nutshell, when an application requests a connection to preset domains or IP addresses the VPN connection is initiated. This works not only with Safari and Mail, but many 3rd party applications. In fact, an application doesn't have to design specifically for this feature. If the developers use the high level CFStream and CFSocketStream APIs iOS will take care of the rest. The main exception is if the developer chooses to use raw socket connections, which do not tie into the VPN on Demand feature (Apple, 2009)

3.6. Configuration Profiles

The most basic security controls are managed through settings on the iOS device. These can either be configured manually, or through configuration profiles that are applied. The profiles are xml files that can customize a wide variety of iOS settings and are created through the *iPhone Configuration Utility* or with a 3rd party tool. For security, the profiles can be signed and/or encrypted to hopefully prevent modification or leaking sensitive information. If a setting is configured through a profile, it cannot be modified directly on the device. The profile can also be set to prevent its removal without a password, or not allow removal at all. We'll cover some of the more valuable profile settings next few sections. For further reading refer to *Apple's Enterprise Deployment*.

3.6.1. Device Passcode

A passcode is the first line of defense against thieves, casual snoopers, or the 3 year-old that gets their hands on your phone. Several options exist for controlling the passcode through a configuration profile as seen in Figure 1. They are all common settings, and can be matched to existing password policies. The default is a 4-digit

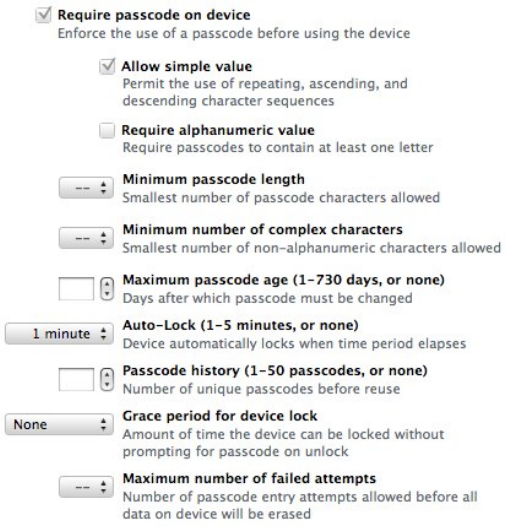

Figure 6

"simple" passcode which is most likely sufficient for many home user's data, but not businesses'. This is simply because of the value of the data, the likelihood of attack versus the in-security of an all digit passcode.

Remember, the passcode will be typed frequently on a small keyboard and inconveniencing the user unduly is not a way for the security team to earn support. Strike a balance between the likelihood of an attack in this manner, and the likelihood of annoying your user-base.

Passcodes are useless if the system the iOS device was sync'd to is available. During forensic examination the passcode can be bypassed with a tool such Lantern from Katana Forensics if the authentication keys are pulled from the computer. The authentication keys are found in the *lockdown* folder of the sync'd machine (Morrissey, 2010, p84). The plist files in the lockdown folder are xml files that contain authentication keys which can be used to circumvent the passcode.

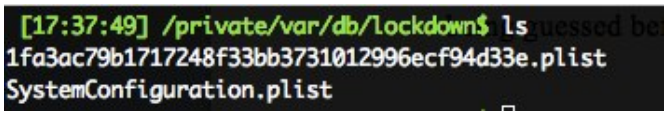

Figure 7

Passcodes are like the locks on car doors – yes they can be bypassed, but they at least raise the bar of difficulty.

3.6.2. Device Restrictions

Device restrictions allow a company to reduce the functionality of the iPhone or iOS device, which in turn reduces the attack profile. It is doubtful that users would agree to have device restrictions applied to their personal devices but it is a good option for organizational owned devices. If the company policy restricts the device to business-only tasks, the following settings should be considered.

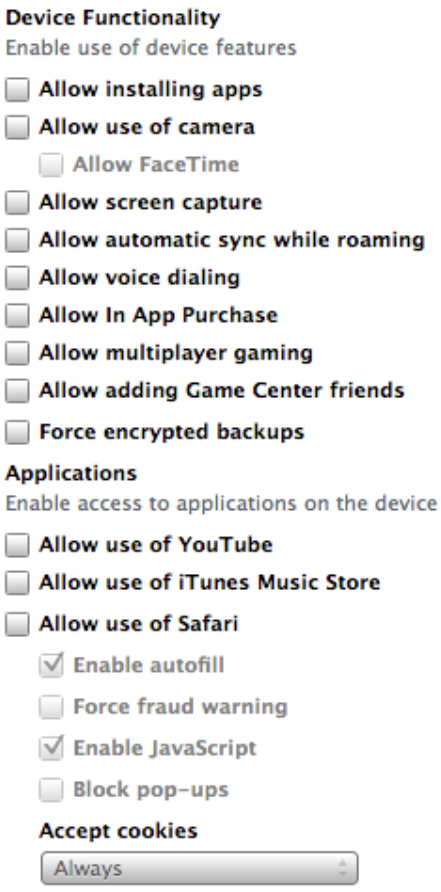

Figure 8

Allow Installing Apps: Uncheck this to prevent users from installing games, personal use applications and other non-white listed tools from being installed. If an application is later needed it can be installed by the IT team.

Allow Use of Camera: If the device will be in an environment that cameras are not allowed it can be disabled.

Allow in App Purchases: If the device is linked to a company managed iTunes account, then it is doubtful the users will need to purchase items inside applications.

Allow Use of Safari: The situations where this is a viable option are pretty minimal. After all, smartphone equals Internet access for many people. However, if you provide access to the needed data through a custom application and the device is used for specific use case you may not need standard web access.

Force encrypted backups: Unlike the other options, there is no reason this restriction should not be in use.

4. Understanding the Implications and Taking Action

Now that we've looked at the threats and some of the mitigations it's time to finalize what the implication to the data is.

4.1. Evaluate

The implications are going to vary from organization to organization, but the first step is to evaluate the data being accessed. This will be based on three questions:

- What data will be accessed
- Who will access it
- How will they access it

The goal is to make sure you understand ahead of time what is needed and to make informed decisions on them. Be as detailed in your answers as possible.

4.2. Classify

Ideally this step is mostly done – your mobile data classifications should match up with existing ones. When classifying your data you are grouping it based on the value or sensitivity, as well as the groups of people that should have access to it. For example, the sales department needs access to customer info and current product inventories. Basic customer info is already stored on normal mobile phones and is acceptable on smartphones. Past sales data for a customer is currently controlled by an internal web application and should remain that way.

Keep the classification levels as minimal and straight forward as possible. Whenever possible make sure the data is marked with its level to remind users. This is not simply a "military" step. All organizations should help their members understand what data is important to keep private, and to what extent.

4.3. Regulate

Regulate in this context refers to having known policies and procedures. These two items are based on the combination of the implications and how you've evaluated and categorized your data. This step means incorporating the mitigations you'll be taking.

4.4. A Word on Personal Devices

Personal devices are an area of contention in a lot of environments. IT groups are reluctant to let a device they can't control have access to their network. Management doesn't want to pay for a device for all the users who want one. And, users don't want to carry two devices. There is not a great solution, but given an option like a custom

application, requiring VPN access, and mandating a release to allow remote wipe a certain level of confidence can be gained.

In November of 2010, NPR published a story of an iPhone user's personal phone being remotely wiped while traveling (Kaste, 2010). She had it synchronized with her company's Exchange system for business email. During her trip the IT department accidently wiped it remotely. It is unclear from the article if her employer had a policy guiding phones being wiped in general, or accessing data on a personal phone. The story serves as an example of what could go wrong. If allowing personal devices to sync with Exchange the policy must clear what actions the IT department can take and what protection is offered to personal data. Organization lawyers should also investigate under what conditions would the company be held liable for loss of data on a personal device.

5. Conclusion: the sky is not falling

There are many risks associated with having data on iOS devices (or any smartphone or tablet), but they are not all that different than laptops. The threats come in a slightly different form, but for the most part are similar. For that reason, the defenses also appear similar. Data encryption, passcodes, network security and user training are the basics of defense – the same as if we were dealing with laptops or netbooks.

In a June article in InfoWorld Robert Lemos makes the bold statement that "Several factors have combined to make the iPhone's and iPad's operating system into what is arguably the most secure commercial OS – desktop or mobile" (Lemos, 2011). While I'm not sure that is entirely accurate, it does reflect the lack of exploitation occurring today. The saying goes, you don't have to out run the bear, you only have to out run the guy next to you. In the security ecosystem this applies - to a point. The majority of attackers will go after the low hanging fruit, and the targets that have the greatest ROI (whether that is money, press, or street cred). Apple has managed with iOS 4 (and it appears more so with iOS 5) to raise the bar necessary to exploit their system. It will get attacked and will be compromised but by keeping the bar high it minimizes the

degree this will occur. This also *decreases* the severity of negatives implications of having the devices in an organization.

While iOS devices themselves are less likely to be compromised, data can still be stolen in-transit and credentials can still be stolen through social engineering. They can be used as a stepping-stone to gain access to the data. The impact of iOS devices will have less to do with overt direct attacks, and more about how users interact with the data itself.

The exact implications to your data on iOS devices are unique, and based greatly on the business or operational impact of its confidentiality or integrity. It is also a decision the security team alone cannot make. Hopefully by reading this paper you have a better understanding of some of the implications, the areas to further research, and some ideas as to how your organization can integrate iOS devices.

6. References

Apple. (2011, March 9). *About the security content of iOS 4.3*. Retrieved April 3, 2011, from Apple Developer Network: http://support.apple.com/kb/HT4564

Apple. (2010, December 1). *Apple Developer Network*. Retrieved January 12, 2011, from About the security content of iOS 4.2: http://support.apple.com/kb/HT4456

Apple. (2010, April). *iPad Security Overview.* Retrieved December 10, 2010, from apple.com: http://images.apple.com/ipad/business/pdf/iPad_Security_Overview.pdf

Apple. (2010, February 24). *The Application Runtime Environment.* Retrieved January 3, 2011, from iOS Developer Library: http://developer.apple.com/library/ios/#documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/RuntimeEnvironment/RuntimeEnvironment.html#//apple_ref/doc/uid/TP40007072-CH2-SW3

Blazaki, D. (2011, January 11). *The Apple Sandbox.* Retrieved June 1, 2011, from Blackhat Media: https://media.blackhat.com/bh-dc-11/Blazakis/BlackHat_DC_2011_Blazakis_Apple_Sandbox-wp.pdf

Dunham, K., Abu-Nimeh, S., Becher, M., Fogie, S., Hernacki, B., Morales, J. A., et al. (2009). *Mobile Malware Attacks and Defense.* Burlington, MA, USA: Syngress Publishing, Inc.

ElcomSoft. (n.d.). *Elcomsoft Phone Password Breaker*. Retrieved May 15, 2011, from Elcomsoft: http://www.elcomsoft.com/eppb.html

European Network and Information Security Agency. (2010, December 10). *Smartphone Security.* Retrieved December 12, 2010, from http://www.enisa.europa.eu/act/application-security/smartphone-security-1

Gartner. (2010, October 21). *Press Releases.* Retrieved November 6, 2010, from Gartner: http://www.gartner.com/it/page.jsp?id=1455314

Kiel Wadner, kiel@timeisswift.com

iphone wiki. (2011, April 29). *N90ap*. Retrieved May 6, 2011, from iPhone Wiki: http://theiphonewiki.com/wiki/index.php?title=N90ap

Kaste, M. (2010, November 22). *Wipeout: When Your Company Kills your iPhone.* Retrieved November 23, 2010, from NPR: http://www.npr.org/2010/11/22/131511381/wipeout-when-your-company-kills-your-iphone

Katalov, V. (2011, May 23). *ElcomSoft Breaks iPhone Encryption, Offers Forensic Access to File System Dumps*. Retrieved May 24, 2011, from Elcomsoft: http://blog.crackpassword.com/2011/05/elcomsoft-breaks-iphone-encryption-offers-forensic-access-to-file-system-dumps/

Kellog, D. (2010, June 4). *nielsenwire.* Retrieved November 6, 2010, from Nielson: http://blog.nielsen.com/nielsenwire/online_mobile/iphone-vs-android/print

Lemos, R. (2011, June 6). *Apple iOS: Why it's the most secure OS, period*. Retrieved June 6, 2011, from InfoWorld: http://www.infoworld.com/d/mobile-technology/apple-ios-why-its-the-most-secure-os-period-792-0

Microsoft. (2006, September 26). *A detailed description of the Data Execution Prevention (DEP) feature in Windows XP Service Pack 2, Windows XP Tablet PC Edition 2005, and Windows Server 2003.* Retrieved June 21, 2011, from Microsoft Support: http://support.microsoft.com/kb/875352/EN-US/

Miller, C. (n.d.). *iOS Exploits.* Retrieved June 22, 2011, from MacDirectory: http://www.macdirectory.com/component/option,com_exclusive_news/task,viewDetail/news_id,3642/

Morrissey, S. (2010). *iOS Forensic Analysis for iPhone, iPad, and iPod touch.* Apress.

Muller, T. (2008, February 17). *ASLR Smack & Laugh Reference.* Retrieved June 2, 2011, from Carnegie Mellon University Personal Webpage: http://www.ece.cmu.edu/~dbrumley/courses/18739c-s11/docs/aslr.pdf

Naraine, R. (2010, March 10). *Charlie Miller wins Pwn2Own again with iPhone 4 exploit*. (ZDNet, Producer, & ZDNet) Retrieved March 15, 2010, from

http://www.zdnet.com/blog/security/charlie-miller-wins-pwn2own-again-with-iphone-4-exploit/8378

Porter Felt, A., & Wagner, D. (2011, June). *Phishing on Mobile Devices.* Retrieved June 22, 2011, from http://w2spconf.com/2011/papers/felt-mobilephishing.pdf

Porter, A. (2010, April 10). Retrieved March 21, 2011, from iPhoneDesk Project Page: http://code.google.com/p/iphonedisk/

Santana, J. (2010, Auguest 9). *New data about the cost of cyber-crime*. (P. Security, Producer) Retrieved June 15, 2011, from Panda Security Insight: http://www.pandainsight.com/en/new-data-about-the-cost-of-cyber-crime

Schwartz, M. J. (2010, November 19). *CIOs See Smartphones As Data Breach Time Bomb*. Retrieved November 20, 2010, from Information Week: http://www.informationweek.com/news/hardware/handheld/showArticle.jhtml?articleID=228300244&cid=RSSfeed_IWK_All#

Seriot, N. (2010, July). *iPhone Privacy.* Retrieved June 11, 2011, from http://seriot.ch/resources/talks_papers/iPhonePrivacy.pdf

TrustedBSD. (n.d.). *Trusted BSD*. Retrieved June 22, 2011, from SEDarwin: http://www.trustedbsd.org/sedarwin.html

Vance, A. (2010, June 24). *Limitations of Data Protection in iOS 4*. Retrieved October 2, 2010, from Anthony Vance - Blog: http://anthonyvance.com/blog/forensics/ios4_data_protection/

Zdziarski, J. (2009, July 24). *Bypassing iPhone 3G[s] Encryption*. Retrieved November 23, 2010, from Jonathan Zdziarski's Domain: http://www.zdziarski.com/blog/?p=516

Last Updated: August 5th, 2020

Upcoming SANS Training

Click here to view a list of all SANS Courses

SANS Reboot - NOVA 2020	Arlington, VAUS	Aug 10, 2020 - Aug 15, 2020	Live Event
SANS FOR508 Sydney August 2020	Sydney, AU	Aug 17, 2020 - Aug 22, 2020	Live Event
SANS Virginia Beach 2020	Virginia Beach, VAUS	Aug 30, 2020 - Sep 04, 2020	Live Event
SANS London September 2020	London, GB	Sep 07, 2020 - Sep 12, 2020	Live Event
SANS Philippines 2020	Manila, PH	Sep 07, 2020 - Sep 19, 2020	Live Event
SANS Baltimore Fall 2020	Baltimore, MDUS	Sep 08, 2020 - Sep 13, 2020	Live Event
SANS Munich September 2020	Munich, DE	Sep 14, 2020 - Sep 19, 2020	Live Event
SANS Network Security 2020	Las Vegas, NVUS	Sep 20, 2020 - Sep 25, 2020	Live Event
SANS Australia Spring 2020	, AU	Sep 21, 2020 - Oct 03, 2020	Live Event
SANS Northern VA - Reston Fall 2020	Reston, VAUS	Sep 28, 2020 - Oct 03, 2020	Live Event
SANS San Antonio Fall 2020	San Antonio, TXUS	Sep 28, 2020 - Oct 03, 2020	Live Event
SANS FOR500 Milan 2020 (In Italian)	Milan, IT	Oct 05, 2020 - Oct 10, 2020	Live Event
SANS Amsterdam October 2020	Amsterdam, NL	Oct 05, 2020 - Oct 10, 2020	Live Event
SANS Brussels October 2020	Brussels, BE	Oct 05, 2020 - Oct 10, 2020	Live Event
SANS Prague October 2020	Prague, CZ	Oct 12, 2020 - Oct 17, 2020	Live Event
SANS London October 2020	London, GB	Oct 12, 2020 - Oct 17, 2020	Live Event
SANS Orlando 2020	Orlando, FLUS	Oct 12, 2020 - Oct 17, 2020	Live Event
SANS October Singapore 2020	Singapore, SG	Oct 12, 2020 - Oct 24, 2020	Live Event
SANS Stockholm October 2020	Stockholm, SE	Oct 19, 2020 - Oct 24, 2020	Live Event
SANS Dallas Fall 2020	Dallas, TXUS	Oct 19, 2020 - Oct 24, 2020	Live Event
SANS Rome October 2020	Rome, IT	Oct 19, 2020 - Oct 24, 2020	Live Event
Cloud & DevOps Security 2020	Denver, COUS	Oct 19, 2020 - Oct 24, 2020	Live Event
SANS SEC504 Rennes 2020 (In French)	Rennes, FR	Oct 19, 2020 - Oct 24, 2020	Live Event
SANS Geneva October 2020	Geneva, CH	Oct 26, 2020 - Oct 31, 2020	Live Event
SANS SEC560 Lille 2020 (In French)	Lille, FR	Oct 26, 2020 - Oct 31, 2020	Live Event
SANS San Francisco Fall 2020	San Francisco, CAUS	Oct 26, 2020 - Oct 31, 2020	Live Event
SANS Cologne October 2020	Cologne, DE	Oct 26, 2020 - Oct 31, 2020	Live Event
SANS Krakow November 2020	Krakow, PL	Nov 02, 2020 - Nov 07, 2020	Live Event
SANS London November 2020	London, GB	Nov 02, 2020 - Nov 07, 2020	Live Event
SANS Rocky Mountain Fall 2020	Denver, COUS	Nov 02, 2020 - Nov 07, 2020	Live Event
SANS DFIRCON 2020	Miami, FLUS	Nov 02, 2020 - Nov 07, 2020	Live Event
SANS Sydney 2020	Sydney, AU	Nov 02, 2020 - Nov 21, 2020	Live Event
SANS OnDemand	OnlineUS	Anytime	Self Paced
SANS SelfStudy	Books & MP3s OnlyUS	Anytime	Self Paced