https://doi.org/10.48009/2_iis_2011_181-193

Volume XII, No. 2, pp 181-193, 2011

VULNERABILITIES AND THREATS TO MOBILE DEVICE SECURITY FROM A PRACTITIONER'S POINT OF VIEW

Joseph Packy Laverty, Robert Morris University, laverty@rmu.edu
Frederick G. Kohun, Robert Morris University, kohun@rmu.edu
David F. Wood, Robert Morris University, wood@rmu.edu
John Turchek, Robert Morris University, turchek @rmu.edu

ABSTRACT

This paper first provides a concise and categorical overview of security vulnerabilities and threats to: a) mobile physical devices, b) mobile operating systems, c) mobile application development, and d) mobile network connectivity and communication derived from the IT practitioner literature. The investigation relies on data and sources used by IT practitioners. It concludes by presenting the resulting "Best Practices" and recommendations for inclusion of Mobile Device Security in the CS/IS curriculum.

Keywords: Mobile Devices, Mobile Device Security, EvDO, Bluetooth, WAP 2.0, SMS, MMS, Location Aware Technologies

INTRODUCTION

The rapid growth in the number, types and functionality of mobile devices has been stunning. At the end of 2010 Dhanjani (2011) reported that over 100 million iPhones were in use, 15 million iPads , and over 10 billion mobile apps were downloaded [9, p.1]. Total global mobile device shipments topped 101 million units in the fourth quarter of 2010 with Android and Symbian-based platforms leading the way, 32.9 million and 31.0 units respectively [5]. The focus of mobile applications has evolved from its origins of email/browsing, entertainment, gaming and social networking to providing remote access to sensitive consumer and enterprise functions. Commercial banks provide mobile web browser access and applications to view account balances, pay bills online, make deposits and transfer funds between accounts [26, 7]. Currently there are over 17,000 health care mobile applications listed in major app stores, of which 50% are directed to health care professionals. The Global Health Care Market Report 2010-2015 estimates that 40% of 1.4 billion Smartphone users will be using health care mobile applications by 2015 [28].

Industry demand and career opportunities in mobile application development and mobile enterprise management has provided both opportunities and challenges for Information System (IS) and Computer Science (CS) curriculums. Mobile application development may be integrated in IS/CS curriculums as an extension of existing of computer program language curricula. Android and Blackberry application development frameworks are an extension of Java. iOS, Symbian and native application frameworks are an extension of C/C++ . Windows Mobile/ Phone 7 provides opportunities to extend the .NET Framework, though support for C#. VB.NET began in late 2010 [33].

While the inclusion of mobile application development concepts within program language curricula may seem obvious, it is also important to consider inclusion of additional security vulnerabilities and threats that mobile applications, operating systems and devices may present to consumers and enterprises. Traditional approaches of IS/CS curriculums have focused security content on three fundamental IT infrastructure components: network services, the host operating system, and applications. This paper first investigates security vulnerabilities and threats to: a) mobile physical devices, b) mobile operating systems, c) mobile application development, and d) mobile network connectivity and communication. The investigation relies on data and sources used by IT practitioners. It continues by presenting "Best Practices" and recommendations for inclusion of Mobile Device Security in the CS/IS curriculum.

METHODOLOGY

This paper provides a different perspective for mobile device security issues in that the data used for this analysis was derived solely from IT practitioner based publications. The identified concerns were derived from experience in responding to reported security issues by IT practitioners. A content analysis was used to identify practitioner defined security vulnerabilities and concerns as they appeared in a convenience sample of IT practitioner publications—a combination of both hardcopy and online. The security vulnerabilities and concerns were then categorized and organized according to frequency and depth of discussion. The results are presented as follows.

MOBILE PHYSICAL DEVICE VULNERABILITIES

Each mobile hardware platform has its own specific set of vulnerabilities. While the hardware capabilities of mobile devices have improved in recent years, mobile devices, e.g., PDAs, Smart phones, etc., continue to have less processing power and internal memory than comparable laptops and desktop units. Such limitations have caused restrictions in using encryption. Older mobile devices do not have the power to use SSL/TLS to secure HTTP transmissions. SSL/TLS is used to prevent Man-in-the-Middle Attacks (lack of certificate match), privacy leaks of data and mobile device ids [10, p. 259]. While replacing a mobile device may at first be an obvious solution, mobile application programmers tend to write code for the lowest common hardware platform. The use of non-SSL forms on mobile HTML is still very common. Sending authentication and sensitive mobile data is especially vulnerable. Some critics argue that 3G and 4G networks are very difficult to sniff making SSL less important. However, older mobile, which are not capable of supporting SSL do not use these types of high-speed CDMA networks [10, pps. 6-7].

Device Physical Limitations

Many mobile devices, e.g., smart phones, have numeric-pad style keyboards and limited display space. The physical limitations of mobile device keyboards present challenges to application and enterprise authentication. Strong password protection strategies recommend a combination of upper-and-lower case letters, numbers and special characters. Many mobile keyboards make enforcement of secure authentication passwords policies very difficult if not impossible [10. p.3]. Jon Heimerl, a director of an enterprise security service provider, describes the vulnerability of touch screen devices that frequently leave finger prints and assists in discovering mobile device passcodes [15].

Device Viewing Space Issues

Limited viewing space of many mobile devices presents additional security vulnerabilities for mobile browsers and email clients. For example, many mobile browsers can not display the full URL in the address bar. This limitation may result in more successful phishing attacks and spoofing attacks, like the 2010 iOS Safari/Skype attack [3, p 272; 1, p. 4]. Many enterprises and organizations have designed mobile web sites have redesigned login pages that address these display limitations. Mobile web site design "Best Practices" are available to promote better usability and security for mobile browsers [10, p13; 1, 14].

Mobile email users are also vulnerable to limited viewing space. Mobile email users tend to rely more on URL links, e.g., responding or navigating in emails. For example, the inability of a mobile email client or browser to display a full URL in links may increase vulnerabilities of a HTTP Redirect attack. Assume that www.mybank.com is legitimate, but www.attacker.com is not. The following carefully constructed HTML link may be inserted into your email message, e.g., http:://www.mybank.com?Login=form.www.attacker.com. The domain name is legitimate, but the crafted URL is passing login data submitted to your bank's application in an attempt to redirect the request to the attacker site. While one would hope that your bank properly validates the data, the mobile user may not see any reference of this phishing scheme displayed in their in their email client or browser [10, pps. 270-271].

Stolen or Lost Devices

The most serious physical security threat is a lost or stolen mobile device. Prolong physical control of a mobile device permits a hacker to use a computer or other methods to compromise local storage of passcodes and encryption keys [15]. Many mobile installed applications, e g.., browsers, email, etc. may also store sensitive data or application authentication tokens in the mobile device's local storage.

Some mobile platforms and service providers provide complimentary services to help locate lost mobile devices. These device location services may require consumer registration or customized web pages to locate and ring active mobile devices [19, p. 15; 16]. Other mobile devices may permit automatic device lock down, automatic content wiping or encryption key erasing if the user fails to successfully enter a device passcode or pin within a specified number of attempts. Any security wiping or erasing policy should be balanced with the quality of an appropriate backup plan and probability of accidental wiping, e.g., children playing with the mobile device [11, 35, 4]. Malicious mobile applications may be wiped by an App Store when malicious behavior has been detected, e.g., DreamDroid Trojan horse [30].

Some mobile platform or service providers provide consumers with backup and remote wiping services at a reasonable cost. Apple's MobileMe service at the cost of $99 per year provides tools for syncing, backing up and securing data, including the ability to sound a tone and/or display a message on a lost iPad if you have temporarily misplaced it [34]. At the enterprise level Microsoft's Exchange Active Sync, Blackberry's Enterprise Servers (BES) and other third-party party solutions provide for a more comprehensive solution for security policies, deployment, updates and protection of loss or stolen devices [3, 34].

MOBILE OPERATING SYSTEM VULNERABILITIES

Even though some mobile operating systems are based on a multitasking/multiuser architecture, mobile operating systems do not support individual user authentication or authorization security. If a user has access to the device pin or passcode, it has access to all local applications and stored data. While local mobile applications may simulate user authentication and authorization on a per-application basis, this security is provided independent of the physical device and operating system. Clarke and Furnell (2005) reported that one-third of the respondents to a survey did not use any device authentication. Even when device passcodes and pins are used, the actual identity of the user may not be determined if multiple users are sharing device passcodes [3, 8]. Mobile enterprise solutions and VPN access to remote user-based storage will provide more secure authentication and authorization services [3, 18].

Local User IDs

Android, Blackberry, and Apple's iOS mobile operation systems adapted the concept of a local user-id to represent an installed mobile application. For example, an Android application and data files are stored, using Linux file permissions, under a /home subdirectory based on the user-id assigned to the application. Android applications may create files and assign other file permissions. Traditional Linux file system permissions are based on user, group and other identities. However, these identities are based on the identity of other applications, not the cell phone user [10, p. 38]. For example, a mobile email client after retrieving email from a POP server would store email messages in a local in-box. All users using the mobile device may have access to the same local email in-box. The application has file permission access, not the user. Other applications may be able to access another application's local data, by appropriately setting file permissions to the application's home directory. If a file is world-writable, a malicious program could miss allocate the memory or use all storage space [10, p. 39].

File and Directory Permission

Other mobile operating systems use file and directory permission similar to Android. Apple and Blackberry use application identity to assign file permissions. Both iOS and Blackberry use a hierarchical file system, but older Blackberry devices have no root directory [10, p.144]. Blackberry Version 7 will change from the Blackberry proprietary operating system to QNX - a Linux-based operating system. SymbianOS and Windows Mobile device storage has no file-level security. Any application can access and other application's data. Windows Mobile, also, does not support a hierarchical storage system [10, p.114, p. 218].

Passwords

Most mobile applications may secure local files by use of a user-set password, a digital signature, or file encryption. In order for an application to access a digitally-signed file, it must provide the file's digital signature. While a file may have any numbers of digital signatures, only one digital signature is required to access the file. Using digital signature to protect sensitive files can not differentiate between individual users.

Summary

To summarize the use of operating system file permissions, a) some mobile operating systems and applications control access to its local files, b) some mobile operating systems and applications local files are always available to other applications (Windows Mobile and SymbianOS), and c) all removable storage is always public. Setting file permission does not always provide the level of protection needed. For example, a recent file permission security flaw in the Android's Skype application meant that your user name, email address, contacts, and pretty much everything else that can be stored by the Skype application could be at risk for data thieves. Permissions were set to permit any installed application to use Skype's mobile stored data [31].

Use file encryption to protect sensitive data when the mobile operating system can not. Blackberry devices may create an encryption key from the lock down password or a randomly generated number stored on the device [10 pps. 146 -147]. By using the lock down password, Blackberry file encryption may be device dependent. Any user or application that has access to the encryption key can use the encrypted file. Given the vulnerabilities of reverse engineering attacks, do not store encryption keys within the application code [19]. Each mobile operating varies in the method used to store encryption keys. When data is stored on removable storage devices, e.g., SDC cards, there are no file system permissions.

MOBILE APPLICATION VULNERABILITIES

Each mobile operating supports different application programming languages. Android and Blackberry's applications are based on the Java programming language, but require specialized mobile APIs. Android APIs are open source; whereas, Blackberry's APIs are proprietary. Apple iOS and Symbian's application programming languages are based on C/C++. Apple iOS uses a proprietary version called Objective C, while Symbian uses Carbide C++. The Windows .NET Framework supports C#.NET and the VB.NET programming languages.

Native Code

In addition to the variety of mobile application programming languages, is the inconsistent support of native code, i.e., applications written in C or C++. Native code will provide better performance for some categories of applications, e.g., gaming. Native code applications are supported by most mobile operating systems. At the time of this writing Windows Phone 7 does not support native code. Mobile native code applications introduce security concerns: a) code is not managed, b) code may compromise the security policy, or c) code may be a target of malicious attackers [29].

Mobile Application Security

Mobile Application Security Models use a combination of application sandboxing, application signing, an installation capability/resource list, and permission model. Most mobile operating systems will execute applications in a sandbox that is designed to isolate each application and security policy so not to cause any problems to the executing environment [22]. Windows Mobile and Apple's iOS doe not use an application sandbox. Both mobile operating systems rely of a rigorous certification process to sign applications to prevent malicious behavior from applications [10, p. 360].

Digital Certificates

Mobile app stores require an application to be signed before it can be distributed. To sign code, you need a digital identity, which is a private cryptographic key plus a digital certificate. You may register your identity with third-party companies such as VeriSign, RSA, or Thawte, and/or using self-signed digital certificates. Blackberry and Apple require each developer to obtain digital certificates only from Research in Motion or Apple [10]. Once a digital certificate has been created, a mobile operating system will import and store digital signatures to verify signed applications. Apple mobile devices use an encrypted certificate storage system called "Keychain", and Android applications use a similar system called a "keystore" [9, pp. 412-419].

Private Keys and Certification

The code developer will use the private key to sign the application code. Signed applications can verify the identity of the application author, provide non-repudiation, protection of intellectual property, and confirm that the original code has not been altered. But digital certificates and code signing, by itself, does not provide protection from malware

Managed Certification

Most mobile application development processes permit self-signed code to be developed and tested using emulators or other mobile devices. However, Blackberry, Apple and iPhone also require a managed certification process to sign application code to be distributed through the app market. This certification process is designed to test for risky applications before they are installed using the app market. For example, the Windows Phone Certification process ensures that Windows mobile applications are: a) reliable, b) makes efficient use of the phone resources, c) does not interfere with the device's functionality, and d) is free of malicious code.

Android Applications

All Android applications must be signed to be installed, but a self-signed Android application may be installed and distributed by Android's App marketplace. The market site for Android requires that you register as an Android developer. All one needs is $25 and a Goggle email account. No designated certificate authority or certification process is required.

Most mobile devices and operating systems provide extra security for third protected applications and resources, e.g., phone, camera, network, etc., that may be used by third-party applications. Each platform provides a different approach to managing secure use of these protected resources. During the installation process an Android application notifies the user the need to access a protected resource, e.g., the phone, and the nature of the protection, e.g., Android Manifest [10, p. 25]. After installation the use of a protected resource by an Android application is not further reviewed.

Confirmation and Capability Lists
Apple uses a similar approach, but requires the user to confirm access to protected resources for each use. While Windows Mobile applications require a "Capability List" to fully disclose the use of protected resources and applications, the Microsoft Signing process is responsible for reviewing malicious behavior during the certification process to access those resources [12, p.436]. Blackberry also provides a certification process to protect platform resources, but it sets or pushes enterprise customized settings for access to a protected resource, either for all applications or by individual applications [13, pp. 72-80].

MOBILE NETWORK CONNECTIVITY AND COMMUNICATION SECURITY

There is a wide variety of cellular physical and data link communication standards. Each standard specifies a frequency band, e.g., GMS, UMTS, and PCS, and a channel access method, e.g., CDMA, TDMA, FDMA, and SDMA. A frequency band will determine the maximum theoretical bandwidth and indirectly the number of access channels. A cellular channel access method represents the technology used to divide the available bandwidth into multiple independent communication sub-channels. General Packet Radio Service (GPRS), an older and widely available cellular standard, was originally based on Frequency Division Multiple Access (FDMA). Global System for Mobile Communications (GSM), an International 2G Standard, is based on Time Division Multiple Access (TDMA).

CDMA
The Telecommunications Industry Association improved previous IS-95 (CDMAOne), cellular CDMA standard to the current CDMA2000 standard. CDMA2000 is a family of cellular communication standards. The CDMA-1xRT specifies CDMA access standards for traditional GPRS-based carrier systems. Evolution-Data Optimized (EvDO) is used for 3G cellular networks, and Long Term Evolution (LTE) is used for 4G cellular networks. All are based on Code Division Multiple Access (CDMA) [24].

Most cell phones support multiple physical and data link standards. For example, a Blackberry may default to EvDO and then automatically revert to GSM or GPRS (1xRT), when EvDO is not available. GSM connectivity provides for roaming in countries outside the United States [13, pp. 83-85,27, 24].

LTE, EvDO, GSM, and GPRS provide for physical and data link encryption services. LTE and EvDO offers the best encryption services. Using the Cellular Authentication and Voice Encryption (CAVE) Algorithm a 128-bit sub-key is generated. Voice and data is then encrypted using the Advanced Encryption Standard (AES) [35]. GPRS provides for confidentiality by using the GPRS- 45 encryption algorithm. While GPRS encryption is not as powerful as LTE/EvDO encryption, it is more powerful than GSM encryption services [28, p. 35].

Wi-Fi
Most mobile devices may access the Internet either by Wi-Fi using traditional TCP/IP and HTTP, or by CDMA/WCDMA using a collection of protocols called WAP (Wireless Application Protocols). WAP is better designed to compensate for limited hardware resources, bandwidth, and battery life of mobile devices. However, WAP-based applications will offer less functionality and reduced security protection. Some examples of WAP-based applications include mobile browsing, email, short message services (SMS), multimedia message service (MMS), and enterprise clients [10].

WAP
For WAP stack utilization, HTML has been modified by the use of Wireless Markup Language (WML) or the newer WAP 2.0 XHTML-MP markup language. HTTP has been modified by the use of Wireless Session Protocol (WSP). TCP/UDP has been modified by the use of the Wireless Datagram Protocol (WDP). IP PDUs are delivered by the Bearer Protocol which is specific to the service provider. Also, when using the WAP stack , SMS and MMS PDUs are delivered to a SIM or Device ID, rather than to a IP address .

WAP Gap

WAP1.0/WTLS is used predominately by older mobile devices and did not provide for end-to-end encryption security. This vulnerability made a mobile device more susceptible to man-in-the-middle attacks. This vulnerability is better known as the "WAP Gap". When a WTLS encrypted message is decrypted from WTLS to clear text and then encrypted to SSL/TLS at a WAP gateway, data is available in clear text to an attacker. Introduced in 2002, WAP 2.0/SSL provided full end-to-end SSL/TLS encryption [10, pp. 259-259]. Since older WAP 1.0 devices do not benefit by EvDO's AES encryption, this WAP Gap presented an increased risk when transmitting sensitive data that is communicated using cellular methods.

XSS, SQL Injection and CSRF Vulnerabilities

Mobile browsers, like traditional browsers, are vulnerable to Cross-site Scripting, SQL Injection and Cross-site Request Forgery attacks when accessing dynamic web site applications. But, mobile browsers do present additional vulnerabilities. One variation of a cross-site scripting (XSS) attacks allows an attacker to steal a victim's session cookie by injecting and executing a script, then impersonating the victim. Many mobile browsers provide limited storage space for cookies, thus permitting attackers to more easily identify the session cookie. Most mobile browsers do not include XSS security features found in standard browsers, e.g., HTTP only flag or support XSS prevention plug-ins [10, p. 255].

SQL Injection

Another popular attack against dynamic web site applications is called SQL injection. An attacker inserts a SQL statement into a form field, cookie, etc. in an attempt to be executed by a SQL database at the web site. While this is a security problem that needs to be addressed by validations at web site applications, the problem with mobile browser occurs when testing for SQL Injection vulnerabilities from a mobile platform. Many web sites will automatically change their appearance or functionality when a mobile browser is detected. This means that a successful penetration test from a standard browser will not guarantee protection for mobile platforms. In addition, the limited bandwidth of CDMA connections may limit the number of test cases [10, p 256-258].

Cross-site Request Forgery Attack

Mobile browser users may browse a sensitive web site, e.g., their bank, while simultaneously visiting a hostile site. A hostile site can also be access by following a hostile link in an email. A Cross-site Request Forgery Attack (CSRF) attempts to execute a transaction originating from a hostile site against the sensitive site, e.g., transfer money between accounts, without the user's knowledge. While this attack is common for all platforms, mobile platforms create same complexity for penetration testing of CSRF vulnerabilities as SQL Injection Attacks. New mobile-based web application testing products have been designed, e.g., Gorilla Logic's FoneMonkey, overcome some of these limitations [23]. A serious question must be addressed concerning the balance between adequate web application security testing for mobile platforms and the race to provide easy-to-use new mobile application services and functions. Is your dynamic transactions equally protected for mobile devices?

SMS and MMS Vulnerabilities

Texting, sharing pictures, downloading applications and other content are perhaps the most appreciated features of a mobile device. Data content is transmitted between mobile users based on either a Short Message Service (SMS) or a Multimedia Message Service (MMS) Protocol Data Unit. Both SMS and MMS are integrated into the WAP standard as a bearer protocol. SMS and MSS PDUs may provide more practical services than user entertainment. SMS messages can provide notification services from carriers or enterprise mobile administration servers, e.g., you have voice mail, your security settings have been changed, install this update, wipe a device, etc. MSS can deliver content, whose size is greater than one PDU, without user intervention [10, pp. 304-309].

SMS

A SMS message is transmitted using a "store and forward" system and the services of a carrier's Short Message Storage Center (SMSC). SMS and MMS are vulnerable to a class of attacks called "Protocol Attacks." The administrative notification, downloading and automatic execution of SMS or MMS content called is called WAP Push, or pushing content. WAP Push attacks introduce a significant new class of attack vectors. [10, pps. 304-312]

WAP

There are three types of WAP messages: a) Service Indication, b) Service Loading, and c) Cache Operations. A Service Indication type will include the URI and a message to describe the service available. The user is notified and will determine when or if the message will be acted upon. Service Indication messages are used by carriers to notify the availability and location of voice mail. On the other hand, a Service Loading WAP push message loads the service and automatically executes the service and maybe without user notification. The type and nature of user notification and approval of SL messages varies by platform, carrier and mobile operating system. Administrative changes of mobile device settings or software updates are examples of Service Loading messages. A WAP Push attack hopes that a mobile device is not configured to reject service identification or loading messages from only "trusted sites." A specially crafted XML-based WAP Push Service Loading message can be sent and executed on a vulnerable device.

MMS
The size of content of MMS message is greater than one PDU, and hence requires multiple PDUs. As a result the target mobile device is provided a MMS notification which contains the URI (location) of the MMS message content, e.g., the picture, audio, application, etc. It is important to repeat that the user may or may not be notified. To prevent malicious attacks most mobile platforms provide an "optional" security setting to accept only trusted SMS and MMS sources.

There is no problem with the MMS notification process. But, do you trust the source of the message? The behavior of the mobile device varies if trusted sources are not required. The following results are possible form an un-trusted source: a) download the content without user notification, b) download the content with user confirmation but no source URI is displayed, c) download the content with user confirmation and the source URI is displayed, and d) download the content with user confirmation and the source URI and phone number/caller id is displayed. While use of a mobile network firewall may provide security protection, WAP Push vulnerabilities operate behind the firewall and not the mobile device. Thus the mobile device is not protected by the firewall [10, pp. 310-321].

Battery Draining and Silent Billing
Battery-draining and silent billing attacks use WAP Push vulnerabilities as a variation of a denial of service attack. Mobile devices are designed to reduce power consumption during idle periods. After a period of inactivity, e.g., 10 seconds, a mobile device will automatically switch from a "talk-mode" to a "stand-by" mode. An attacker can send continuous MMS notifications or send UDP packets to ping the victim's phone [31]. On the other hand, silent billing attacks bombard the targeted device with MMS messages. Since MMS messages operate in the background and may not be visible to user, a mobile device may accumulate excessive mobile charges [10, p.318].

Over The Air Attacks
Over The Air (OTA) attacks uses the capability of WAP Push technologies to change the settings of a mobile phone. One strategy is design to change the location of the proxy server used by the web browser. This means than a malicious user can intercept your web content using a man-in-the-middle attack. Some devices or security applications may block any OTA message that did not originate from your own network system enterprise administration server, or to redirect WAP Push message to an Anti-Phishing system which can perform analysis of the URL in the WAP Push message [10, p. 318].

Location- Awareness Technology Vulnerabilities
There are a variety of Location-Awareness technologies, e.g., GPS, Assisted GPS, 802.11(WiFi) access point database lookups, or tower triangulation. GPS provides for the greatest degree of location precision, while also providing continuous tracking updates necessary to support real-time location applications. However, GPS location information can take several minutes to acquire and is impeded indoors. “Assisted GPS” attempts to combine GPS location technologies with other technologies to offset these disadvantages [10, pp. 332-334].

Skyhook Wireless
Skyhook Wireless, a WIFI Positioning System (WPS), is based on an 802.11 database containing location data from more than 250 million wireless access points. It is faster but less precise than GPS. But, WPS is more precise than cell tower triangulation, e.g., 20-30m. [10, pps 333-334; 28]. However, when wireless access points are moved, the location data stored in the 802.11 database may be inaccurate. Skyhook Wireless and Google's "Latitude" provide 802.11 databases and application APIs to integrate WPS into mobile applications. Most mobile devices use a combination of all four location positing technologies.

Geo-location
Each mobile platform treats geo-location security differently. For example, each Android location-aware mobile application must get user approval during installation. iOS location-aware mobile applications, on the other hand, by

default must get approval every time the user starts a location-aware application. Windows Phone 7 has no application-by-application user approval system. Windows All installed location-aware mobile applications are allowed to access location data [10, pp. 332-340].

Location-Awareness security raises a new set of concerns and potential liability. In a report submitted to Congress, John Morris (2010) summarized the increased risks of malicious users, companies, and governments abusing the rights and privacy of location-aware mobile device users. While legal prescriptions in the US for the use of location data are limited, malicious access to location data provides opportunities for stalking and domestic violence. In the UK, the Data Protection Act requires that users are made aware of who is tracking them, and the use and retention of location data. When minors are involved, malicious access to location data may pose additional liability problems for app and service providers. Storage of non-anonymous positional data has been a new target for divorce court subpoenas used to provide evidence for adulterous behavior [10, p 340, 28].

Bluetooth Vulnerabilities

The use of Bluetooth technology in mobile devices platforms has continued to grow. The use of mobile headsets is the most common. Other popular uses include; a) connectivity to wireless keyboards and printers, mobile device synchronization with a desktop computer, filer transfer, tethering for laptop Internet access (uses Bluetooth as a modem), and connectivity for hands-free and voice activation services [10].

Pairing

The process of establishing a communication link between a Bluetooth and a mobile device is called "Pairing." The method used to "Pair" devices in Personal Area Network (PAN) is critical to the overall security of the connection. You need to pair Bluetooth devices only once. During the pairing process the devices "agree on and generate keys that are used to identify and reference relationships with other devices" [10, p.288]. Bluetooth v2.1 + EDR introduced the Secure Simple Pairing (SSP) process in 2007. SSP provides multiple methods to pair Bluetooth devices and introduces the ECDH (Elliptic Curve Diffie-Hellman) algorithm to generate and securely exchange link keys. Bluetooth v2.1 + EDR pairing methods may be based on passkeys, numeric comparisons, just works, and out-of-band association models [10, pp. 289-290]. While Bluetooth v2.1+ EDRs has improved authentication, authorization and confidentiality, Bluetooth devices do not provide for data integrity and non-repudiation.

Range

While some consider the limited communication range, e.g., 30 to 330 feet, or the 79 communication channels as security advantages for Bluetooth devices, high-gain antennas and frequency scanning attacks can easily defeat these features. The use of Bluetooth keyboards are targets for keyboard logging attacks and phone conversations can be easily intercepted between a mobile device and Bluetooth headsets. Bluetooth and other methods used to synchronize content between a mobile device and a local computer may be intercepted and provides an attack vector to propagate malware. Bluetooth attack software is readily available. Bluestumbler is a program that can sniff Bluetooth traffic. Bluesnarf can remove data stored on a Bluetooth device, and Bluebrowse can search for Bluetooth device services [28, p.88].

Synchronization

Special consideration should be given to synchronization of content between mobile devices and PCs. Enterprise PCs are often protected by multiple firewall layers, layered malware protection, e.g., email, file, attachments, web traffic, macros, etc, and secure file access. But, during the synchronization process an unprotected mobile device will bypass many enterprise protections and malware may be copied to the enterprise PC. Malicious content that could not be transferred by a remote attack can now be transferred during the synchronization process.

MOBILE DEVICE SECURITY BEST PRACTICES

Physical Device and Operating System Best Practices [30, 10, 34]

1. Use passcodes, passwords and pins to automatically lockdown your mobile device during periods of inactivity.
2. Understand the limitations of lockdowns. Consider the use of an automatic local or remote device wiping service to prevent sensitive data from being compromised after prolong physical control by a hacker.
3. Use encryption services to protect sensitive files, e.g., email inbox, contacts, certificates, documents, etc., either stored on the local mobile device or removable storage.
4. Keep you device firmware, operating system and applications patched and updated.

5. Do not rely on older mobile hardware and operating system platforms to process sensitive applications or store sensitive data.
6. Do not alter a device to provide "root" access, e.g., Android, or "jailbreak" a device, e.g., Apple. These strategies significantly decrease the platform's security protection.
7. Use and automatic update mobile antivirus protection, e.g., Symantec Antivirus for Handhelds, Trend Micro Mobile Security, McAfee VirusScan Mobile, etc. Worms, viruses and malware can be spread through SMS (text messaging), MMS (picture sharing), and Bluetooth device synchronization.
8. Do not provide other users with pins, passcodes, passwords, etc.

Mobile Application Best Practices [10, 28, 30, 19, 13]

1. Stick to official app markets or enterprise-signed application code. Most app stores require applications to pass various secure programming tests to prevent malicious code.
2. Carefully confirm which certificates to add to a Keychain, Keystore, or Certificate store. Certificates confirm identity of an application, not guarantee security.
3. Deny other mobile applications access to certificate stores.
4. Carefully review all application installation and use security notifications. Many mobile application require user consent during installation or use when third-party application require access to installed applications and resources, e.g., camera, phone, Internet, contact managers, email clients, removable storage, etc.

Mobile Browser, Email, Personal Information Content Best Practices [10, 25, 19, 13]

1. Carefully review all URLs and links. Limited mobile device display size increases risks for phishing, HTTP redirects, and Cross-site Request Forgery (CSRF) attacks.
2. Use web sites that are designed and tested for secured mobile device access (not just for standard browsers).
3. Logoff immediately after using a Web application to prevent XSS attacks.
4. Do not allow your browser to save username/passwords, and do not allow sites to "remember" your login password.
5. Do not use the same browser to access sensitive applications and to surf the Internet freely (tabbed browsing). This will help prevent XSS and CRSF attacks against unprotected dynamic web site applications.
6. Encrypt email inboxes and deny other third-party applications access.
7. Deny other third-party applications access to calendars, task, memos and contacts. If possible encrypt and backup the content.

Mobile Geo-location Best Practices [10]

1. Understand the risks associated with location-aware applications and technologies, e.g., ability to track location, evidence to be used in court, etc.
2. Avoid using Geo-location apps that do not provide notification for privacy, sharing and retention of positional data.
3. Avoid providing user information to location-aware applications, e.g., location data should be anonymous.
4. Applications should notify the user when location services are active.
5. Location-aware applications should permit the user to temporarily disable the service.

Networking, Wi-Fi, and Bluetooth Best Practices [10, p. 297, 28, p.87, 30]

1. Use network or messaging firewalls.
2. Use SSL/TLS encryption.
3. Disable WiFi and Bluetooth when not in use.
4. Disable the discoverable mode of Bluetooth devices.
5. Use complex Pins, Passphrases , maximum key sizes to pair sensitive Bluetooth devices in a secured environment
6. Enable mutual authentication
7. Unpair Bluetooth devices that have become lost or stolen
8. If possible, specify trusted SMS and MMS sources and require user approval of all untrusted sources.

Mobile Enterprise Best Practices [19, 13, 10]

1. Control which mobile devices can connect to your Enterprise PC
2. Do not sync mobile devices and Enterprise PCs devices without virus and security protections on both sides.
3. Use VPN to secure access to Enterprise resources, e.g., corporate email, files, etc.
4. Create and Communicate a Data and Mobile Use Policy, e.g., data that may be copied from or to the enterprise network.
5. Enforce Security Policies with available enterprise technologies, e.g., Digital Guard. Signing a policy is not enough.
6. Mobile Enterprise servers require significant security preparation and provision for continuous updates and patches.
7. If Enterprise users have mobile devices and synchronizes their devices with Enterprise PCs, a mobile Enterprise policy and security solution is very important to be considered.

RECOMMENDATIONS AND CONCLUSIONS

The use of mobile devices, iPads, PDAs, and cell phones is pervasive and ubiquitous to consumers, the enterprise, and even to the faculty. Inclusion of mobile device technologies within the IS/CS curriculum can be implemented from a hardware, operating system, application development, or security perspective. A mobile operating system approach offers little additional content value as compared to existing operating system courses. Essentially, mobile operating systems are modified and water-downed from popular operating systems or are proprietary.

IS/CS curriculums have always included application development concepts. Mobile applications are extensions of existing program languages, e.g., C, C++, Java, .NET. Mobile application development can be presented anywhere in the curriculum, e.g., entry-level to capstone; but, will be limited to availability of faculty to teach mobile applications [16]. However, given the interest of students for mobile technologies one should give serious consideration to inclusion of mobile programming languages in an entry-level course – perhaps a more attractive and practical alternative than approaches like Alice.

This papers overviews the vulnerabilities and threats of mobile devices. Most of the "Best Practices" are consumer-based, easy-to-understand and implement. (See Summary Table) Given the increased use, functionality, and importance to consumers and industry, mobile device security should be included throughout the IS/CS curriculum, and not limited to application development courses. From introductory IS/CS survey courses to inclusion in network data communication, operating systems, programming languages, and security curricula, presentation of basic concepts of mobile device and mobile application security is as important to individuals as to the enterprise.

REFERENCES

1 10 Tips for Designing Mobile Websites. (n.d.). Retrieved from *http://labs.thesedays.com/2010/07/16/10-tips-for-designing-mobile-websites/*

2 AT&T FamilyMap. (n.d.). Retrieved on 4/7/11 from: *https://familymap.wireless.att.com/finder-att-family/help.ht*m

3 Blackberry Enterprise Server – Security Technical Overview. (n.d.). Retrieved on 3/29/11 from: *http://docs.blackberry.com/en/admin/deliverables/16650/BlackBerry_Enterprise_Server-Security_Technical_Overview--1153051-0615043613-001-5.0.2-US.pdf*

4 BlackBerry users get free remote wipe, backup and location. (n.d.). Retrieved on 4/2/11 from: *http://www.theregister.co.uk/2011/03/22/blackberry_protect/*

5 Canayls (2011). Google's Android becomes the world's leading smart phone platform. Retrieved on 4/4/11 from: *http://www.canalys.com/pr/2011/r2011013.html*

7 Chase Mobile Applications. (n.d.). Retrieved ion 4/4/11 from: *https://www.chase.com/online/services/mobile-banking.htm*

8 Clarke, N. and Furnell. (2005). Authentication of users on mobile telephones – A survey of attitudes and practices. Retrieved on 2/21/11 from: *http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6V8G-4H9PMXR-1&_user=2144460&_origUdi=B6VNT-4T9DCS5-C&_fmt=high&_coverDate=10%2F31%2F2005&_rdoc=1&_orig=article&_origin=article&_zone=related_art&_acct=C000056294&_version=1&_urlVersion=0&_userid=2144460&md5=701be690707b78c4b3ade36cac13f619*

9 Dhanjani, N. (2011) New Age Application Attacks Against Apple's iOS and Countermeasures. Blackhat Barcelona 2011. Retrieved on 2/27/11 from: *https://media.blackhat.com/bh-eu-11/Nitesh_Dhanjani/BlackHat_EU_2011_Dhanjani_Attacks_Against_Apples_iOS-WP.pdf*

10 Dwivedi, H., Clark, C. and Thiel, D. (2010). *Mobile Application Security*. McGraw-Hill.

11 Google brings remote wipe, other security features to Android through Google Apps dashboard. (n.d.). Retrieved on 4/15/11 from: *http://thenextweb.com/google/2010/10/29/google-brings-remote-wipe-other-security-features-to-android-through-google-apps-dashboard/*

12 Henry, L. Chuvyrov., E. (2010). *Beginning Mobile 7 Development*. Apress

13 Hoffman, D. (2007). *Blackjacking: Security Threats to Blackberry Devices, PDAs, and cell Phones in the Enterprise*. Wiley Publishing, Inc.

14 How to Design and Build a Mobile Website. WebPage FX. (n.d.). Retrieved from *http://www.webpagefx.com/design-build-mobile-web-site.html*

15 Kirk, J. (2010). AT&T apologizes, blames hackers for iPad e-mail breach Computer World. Retrieved on 5/2/11 from: *http://www.computerworld.com/s/article/9178024/AT_T_apologizes_blames_hackers_for_iPad_e_mail_breach*

16 Mahmoud, Q. Teaching Mobile Computing to Generation C. (2008). Sun Developer Network. Retrieved on 4/4/11 from: *http://java.sun.com/developer/technicalArticles/Interviews/community/mahmoud_qa.html*

17 Millan, W., Gauravaram, P. (2004). Cryptanalysis of the Cellular and Voice Encryption Algorithm. Cellular Authentication and Voice Encryption. Retrieved on 5/5/11 from: *http://www.jstage.jst.go.jp/article/elex/1/15/453/_pdf*

18 Mobile Device Security: Securing the Handheld, Securing the Enterprise. (n.d.). Retrieved on 2/21/11 from: *http://www.good.com/media/pdf/enterprise/mobile_device_security_wp.pdf*

19 Mobile Encryption. (2010). Retrieved on 4/2/11 from: *http://msdn.microsoft.com/en-us/library/bb416357.aspx*

21 MobileMe Find Your Phone or iPad. (n.d.). Retrieved on 4/7/11 from: *http://www.apple.com/mobileme/features/find-my-iphone.ht*ml

22 Morris, J. (2010) The Privacy Implications Of Commercial Location-Based Services Retrieved on 4/2/11 from: *http://www.cdt.org/files/pdfs/CDT-MorrisLocationTestimony.pdf*

23 OrrilaLogic. (n.d.). Retrieved on 4/28/11 from: *http://www.gorillalogic.com/what-we-do/mobile-iphone-services*

24 Overview of CDMA2000 Standards. (n.d.). Retrieved on 5/5/11 from: *http://www.radio-electronics.com/info/cellulartelecomms/3gpp2/cdma2000-1xrtt-basics-tutorial.php*

25 Petefish, P. Sheridan, E., Wichers, D. (n.d.). Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet. OSWAP. Retrieved on 5/2/11 from: *https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet#No_Cross-Site_Scripting_.28XSS.29_Vulnerabilities*

26 PNC Virtual Mobile Wallet (n.d.). Retrieved on 4/4/11 from: *https://www.pncvirtualwallet.com/virtual_wallet_mobile.html*

27 Racic, R., Ma, D, Chen, H. (n.d.). Exploiting MMS Vulnerabilities to Stealthily. Retrieved on 4/28/11 from: *http://www.cs.ucdavis.edu/~hchen/paper/securecomm06.pdf*

28 Research2Guidance. (2010). 500m People will be using mobile health care applications in 2015/. Retrieved on 4/4/11 from: *http://www.research2guidance.com/500m-people-will-be-using-healthcare-mobile-applications-in-2015/*

29 Rubin, A., Geer, D. (1998). Mobile Code Security. AT&T Labs Research Retrieved on 4/30/11 from: *http://avirubin.com/ieee.ic.pdf*

30 Saccoi, A., (2011). Android Security: Six Tips to Protect Your Google Phone. Retrieved on 3/23/11 from: *http://www.pcworld.com/article/221650/android_security_six_tips_to_protect_your_google_phone.html#tk.mod_rel*

31 Signal News. (2011). Skype Security Risks Android User Data. Retrieved on 4/2/11 from: *http://signalnews.com/skype-security-312*

33 TechRepublic. (2010). Programming news: VB.NET for Windows Phone 7, Mono 2.8.1, NetBeans IDE 7.0 Beta. Retrieved ion 12/2/10 from: *http://www.techrepublic.com/blog/programming-and-development/programming-news-vbnet-for-windows-phone-7-mono-281-netbeans-ide-70-beta/3424*

34 Wagner, M. (2010). 9 tips for securing your iPad. Computer World. Retrieved on 2/9/11 from: *http://blogs.computerworld.com/16318/secure_ipad*
35 Wagner, W. (n.d.). 9 tips for securing your iPad. Retrieved on 4/15/11 from: *http://blogs.computerworld.com/16318/secure_ipad*

Summary Table	
Mobile Device Vulnerabilities	**Mobile Device Security Best Practices**
Mobile Physical Device Vulnerabilities • Power to Send SSL forms • Threat of lost or stolen device • Physical limitations of keyboards and display challenge authentication an open up to redirect attacks • Limited URL display –opens up to phishing attack	**Physical Device and Operating System** • Use passcodes and passwords • Use encryption services to protect files • Update operating systems and firmware • Use mobile anti-virus protection when available
Mobile Operating System Vulnerabilities • No support for individual user or authentication security • Setting file permissions does not always provide the level of protection needed	**Mobile Applications** • Use official app market apps • Use Certificates • Review installation and security notices
Mobile Application Vulnerabilities • Mobile native code application security concerns - Code is not managed - May compromise the security policy - May be the target of malicious attackers • Digital certificates and code signing do not provide protection from malware	**Mobile Browser, Email and Personal Information Content** • Do not allow browser to save username/passwords • Only use mobile device secured websites • Logoff immediately after using a web application
Mobile Network Connectivity and Communication Security Vulnerabilities • WAP Gap—WAP used by older devices does not provide end-to-end security • XSS,SQL Injection and CSRF mobile browser web site attack vulnerabilities • SMS and MMS vulnerabilities involving Push attacks and message size based malicious attacks behind the firewall—not the mobile device • Location Awareness Technology vulnerabilities open up to malicious assess • Bluetooth vulnerabilities particularly through he transfer of malicious content during device synchronization	**Networking, Wi-Fi and Bluetooth** • Use network or messaging firewalls • Use SSL/TLS encryption • Disable Wi-Fi and Blue tooth when not in use • Disable discoverable mode of Bluetooth • Enable mutual authentication
	Mobile Geo-location • Location data should be anonymous • Applications should notify user when location services are active
	Mobile Enterprise • Do not sync mobile devices and enterprise PC devices without virus and security protection • Enforce security policies with available technologies