International Journal of Research in Advent Technology, Vol.4, No.3, March 2016
E-ISSN: 2321-9637
Available online at www.ijrat.org

Security Guidance for Critical Areas of Mobile Computing

Hemant G Sarode; Prof. Sonal Bankar
Department of Computer Engineering,
Lokmanya Tilak College Of Engineering, Navi Mumbai, India

Abstract: The radical evolution of computer technology particularly in the Hardware (towards reduced size & weight, lower power consumption, higher performance at lower price.) and Communications (wireless & satellite networks, cellular telephony, WAN’s, Internet), has introduced the concept of Mobile Computing. It gives freedom for users don't have to be tethered on expensive wired workstations in order to exchange data. All they need is mobile computers that are portable computers communicating via wireless networks.
The benefits of on-the-move network connectivity are obvious but there are several serious networking & system issues to be solved before the full benefits of mobile computing systems are realized into the practice. Out of these issues one most important issue is Security. The approach of this document is to discuss the security issues generating from the today’s technological changes in mobile computing. Truly mobile computing offers many advantages. Confident access to the Internet anytime, anywhere will help free us from the ties that bind us to our desktops. Having the Internet available to us as we move will give us the tools to build new computing environments wherever we go.
However, there are still some technical obstacles that must be overcome before mobile networkings can be-come widespread. The most fundamental is the security management, which is almost an afterthought until the recent years. Providing security services in the mobile computing environment is challenging because it is more vulnerable for intrusion and eavesdropping.
Keywords: mobile computing, security; threats; attacks; smartphone.

1. INTRODUCTION:

Mobile Computing is very broad term which can be used to define any means of using a computer outside the workplace. This includes working from home or on the road, at airport or at hotel. This also includes kiosks used to remotely connect to corporate office, home computers, laptops, smart phones or tablets. In this paper we have restricts our scope up to mobile devices like smart phones & Tablets. It is interaction between human and computer by which a computer is expected to be transported during normal usage.
The birth of "mobile computing" has signalled a new era in the field of computing and information systems. The concept of mobile computing is derived from the realization that as computing machinery decrease in size and increase in computing power users will demand these machinery to be part of their everyday life,accompanying them in the carrying-out of their everyday tasks. Mobile computing is distributed computing that involves elements whose location changes in the course of computation. Elements may be software components such as mobile agents data, hardware such as palmtops and wireless phones or users The term mobile computing is very often used for wireless mobile computing - the use of portable devices capable of wireless networking.

Wireless mobile computing faces additional constraints induced by the characteristics of wireless communications and the demand for portability. Mobile wireless computing enables access to data at any time and from any place towards the vision of ubiquitous or pervasive computing. Although mobile computing covers a variety of different hardware and software platforms as well as diverse applications, many common issues arise. Mobile computing is human–computer interaction by which a computer is expected to be transported during normal usage. Mobile computing involves mobile communication, mobile hardware, and mobile software. Communication issues include ad hoc and infrastructure networks as well as communication properties, protocols, data formats and concrete technologies. Mobile software deals with the characteristics and requirements of mobile applications.

2. CHARACTERISTICS-MOBILE COMPUTING:

2.1 Portability:

As the name “Mobile” implies, the device is to be able to move from one place to another place without affecting its ongoing functionality. The portability provides the user to take away its digital devices from his/her office location & provides easy access of its working files on the go.

2.2 Connectivity:

The ease of being able to connect to the Internet and receive or transmit data is an essential component to mobile computing. Connectivity through mobile carriers over a 3G- or 4G-type network, as well as Wi-Fi capabilities, are basic requirements for mobile devices.

2.3 Interactivity:

This could almost go without saying, but like most other computing technologies, the ability for a mobile device is critical. The interactivity becomes more significant with mobile devices, as they typically have less computing power than other types of technology.

2.4 Individuality:

Individuality may sometimes be overlooked, but it is a basic component of the concept of mobile computing. Mobile devices, including smart phones and tablets, are designed for individuals and have become a sort of extension to people in many aspects of their lives. From this perspective, how individuals interact with mobile devices remains unique.

3. THREATS TO MOBILE COMPUTING:

Times have changed dramatically since 1946 when the first mobile telephone call was made. For the first 60 years, there was really only one purpose for a mobile device to conduct phone calls. This was a relatively simple process, and for the most part it was secure. Mobile carriers only had to worry about potential phone fraud but security was not something that was high priority for them. Over the past few years, there has been a wave of new mobile devices that run on 3G or 4G networks. These devices include smart devices, PC cards/dongles, and netbooks. In many ways, these new 3G or 4G devices are comparable to today's laptops or desktops, only a lot more mobile. While mobile devices are primarily used for voice communications, they are designed for much more as shown in Figure 1. Today, a typical smart phone is able to: [7]

• Use 3G technology or Wi-Fi to access the data network at broadband speeds.
• Access any website in an open environment.
• Access thousands of mobile applications.
• Synchronize emails, contacts, calendars, etc. with personal and corporate email systems.
• Download and manage digital music, photos, podcasts, videos, and other multimedia.

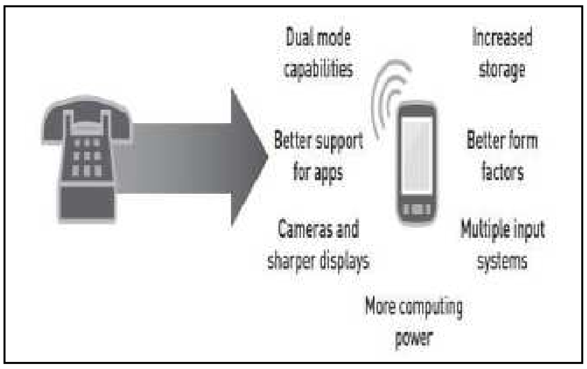

Fig1. Mobile devices are evolving into Multifunctional devices

In July 2012, the Cloud Security Alliance and the Mobile Working Group surveyed 210 security practitioners from 26 countries. Respondents were approximately 80% "experts in the field of information security," which includes security admins, consultants and cloud architects.[3] The survey asked users to rank mobile top threats in order of both their concern and likelihood of a threat: occurring this year, next year, or not likely to happen. After considering over 40 different top threats to the mobile landscape, the top candidates were dubbed "The Evil 8."

3.1 The Evil 8: Top Threats to Mobile:

1. Data Loss from lost, stolen, or decommissioned devices:

By their nature, mobile devices are with us everywhere we go. The information accessed through the device means that theft or loss of a mobile device has immediate consequences. Additionally, weak password access, no passwords, and little or no encryption can lead to data leakage on the devices. Users may also sell or discard devices without understanding the risk to their data. [3]

2. Information stealing mobile malware:

Malware is software that is designed to engage in malicious behavior on a device. For example, malware can commonly perform actions without a user's knowledge, such as making charges to the user's phone bill, sending unsolicited messages to the user's contact list, or remotely giving an attacker control over a device. Malware can also be used to steal personal information from a mobile device that could result in identity theft or financial fraud. [14]

Repackaging is a very common tactic, in which a malware writer takes a legitimate application, modifies it to include malicious code, then republishes it to the app market or download site as shown in the below figure.[15]

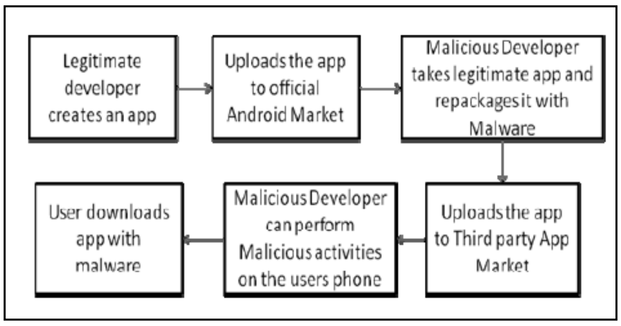
Legitimate developer creates an app
Uploads the app to official Android Market
Malicious Developer takes legitimate app and repackages it with Malware
Uploads the app to Third party App Market
Malicious Developer can perform Malicious activities on the users phone
User downloads app with malware

Fig 2. Process of third party app stores

3. Data Loss and Data Leakage through poorly written third-party applications:

Applications for smartphones and tablets have grown exponentially on iOS and Android. Although the main marketplaces have security checks, certain data collection processes are of questionable necessity; all too often, applications either ask for too much access to data or simply gather more data than they need or otherwise advertise. [3]

4. Vulnerabilities within devices, OS, design, and third-party applications:

Vulnerabilities that can be exploited for malicious purposes. Such vulnerabilities can often allow an attacker to access sensitive information, perform undesirable actions, stop a service from functioning correctly, automatically download additional apps, or otherwise engage in undesirable behavior. Vulnerable applications are typically fixed by an update from the developer [2].

5. Unsecured Wi-Fi, network access, and rogue access points:

The number of locations that provide Wi-Fi in particular, free Wi-Fi has exploded over the last few years. This has increased the attack surface for users who connect to these networks. In the last year, there has been a proliferation of attacks on hotel networks, and open rogue access points installed on public places.

Increased access to public Wi-Fi, along with increased use of mobile devices, creates a heighlightened opportunity for abuse of this connection. Firefox's Firesheep extension is a perfect example of how one can gain access to data through public unsecured Wi-Fi.

6. Unsecured or rogue marketplaces:

Android devices, offers many options for application downloads and installations. Android users can easily opt to download and install apps from third-party marketplaces other than Google's official "Play Store" marketplace. The majority of malicious code distributed for Android has been distributed through third-party app stores. Most of the malware distributed through third-party stores has been designed to steal data from the host device. Tigerbot is downloaded involuntarily to devices from third-party marketplaces. TigerBot is a bot designed to gather confidential data from a mobile device and uses SMS to control the installed bot. In figure below the TigerBot malware hides from the user by masking itself as a popular icon, such as Google's search app, and a generic application name (ie. "System")

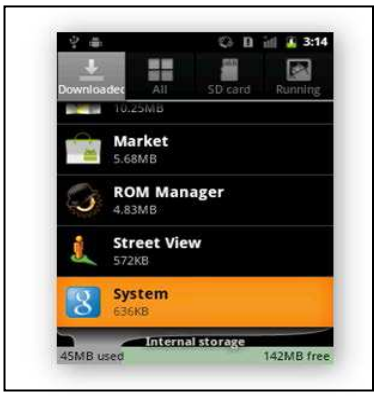

Figure 3. Tiger Bot.

7. Insufficient access to APIs:

Granting users and developers access to a device's low-level functions is a double-edged sword, as attackers, in theory, could also gain access to those functions. However, a lack of access to system-level functions to trusted developers could lead to insufficient security[14]

8. Proximity-based hacking:

Near-field communication (NFC) allows mobile devices to communicate with other devices through short-range wireless technology. NFC technology has been used in payment transactions, social media, coupon delivery, and contact information sharing.[3] Due to the information value being transmitted, this is likely to be a target of attackers in the future.

4. SECURITY SCHEMES IN MOBILE COMPUTING:

As mentioned in section 3 how the malicious code takes acces of the mobile device & steels the highly sensible information from the device. To detect these there are various repackaging detection algorithms available. All these algorithms help to identify a repackaged application on a third party app market. One such algorithm is the DroidMOSS algorithm[4].

Fig.4 shows an overview of DroidMOSS. DroidMOSS has three key steps: Feature Extraction, Finger Print Generation and Similarity scoring. [1]

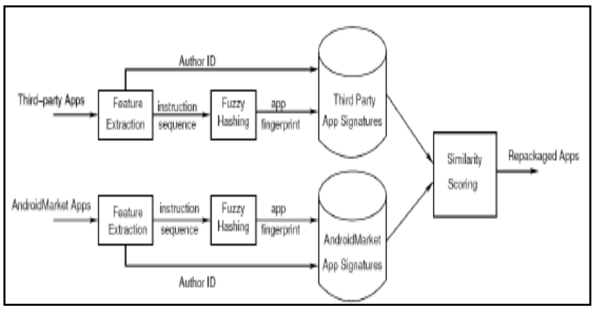

Fig 4. An Overview of DroidMOSS

a) Feature Extraction:

This is the first step where the two main features of each app, that is, instructions contained in the app and its author information, are extracted. These two features are used to uniquely identify each app. Each Android app is essentially a compressed archive file which contains the classes.dex file and a META-INF subdirectory. The classes.dex file contains the actual Dalvik bytecode for execution while the META-INF subdirectory contains the author information. Dalvik disassemblers are used to extract Dalvik bytecode from classes.dex. [10] The code contains opcodes and operands. Further abstraction is made by removing the operands and retaining only the opcode with the believe that it might be easy for repackagers to modify or rename the operands, but much harder to change the actual instructions. For the author information, the META-INF subdirectory contains the full developer certificate, from which the developer name, contact and organization information, as well as the public key fingerprints are obtained. Each developer certificate is mapped into one unique 32-bit identifier (author ID) which is integrated into signature for comparison.[10]

b) Fingerprint Generation:

The second step is to generate a fingerprint for each app, using a specialized hashing technique called fuzzy hashing [4]. Instead of directly processing or comparing the entire (long) instruction sequences, it first condenses each sequence into one much shorter fingerprint. The similarity between two apps is then calculated based on the shorter fingerprints, not the original sequences. The instruction sequence is first divided into smaller pieces. Each piece is considered as an independent unit to contribute to the final fingerprint. However, the challenge lies on the determination of the boundary of each piece. In Droid MOSS, a sliding window is used, that starts from the very beginning of the instruction sequence and moves forward until its rolling hashing value equals a pre-selected reset point, which determines the boundary of the current piece. Specifically, if a reset point is reached, a new piece should be started. [4] The concrete process is presented in Algorithm 1. & visually summarized in Figure 5.

Algorithm 1:

```
Generate the app fingerprint [4]
Input: Instruction sequence iseq of the app
Output: Fingerprint fp
Description: wsize - sliding window size, rp - reset point value,
sw - content in sliding window, ph - the piece hash
1: set_wsize(wsize)
2: set_resetpoint(rp)
3: init_sliding_window(sw)
4: init_piece_hash(ph)
5: for all byte d from iseq do
6: update_sliding_window(sw, d)
7: rh rolling_hash(sw)
8: update_piece_hash(ph, d)
9: if rh = rp then
10: fp concatenate(fp, ph)
11: init_piece_hash(ph)
12: end if
13: end for
14: return fp
```

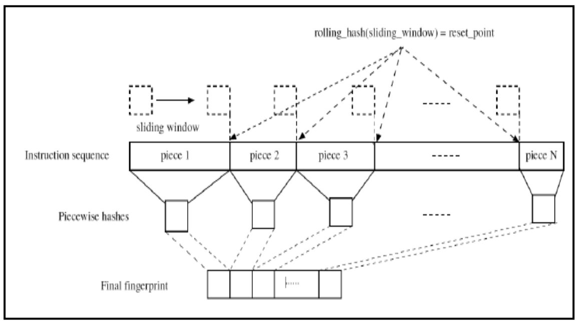

Fig 5. Fuzzy Hashing for Fingerprint Generation

c) Similarity Scoring:

In the third step, divide the apps into two groups, one from the official Android Market and one from alterna-

tive marketplaces, and then calculate pairwise similarity scores between the two. The similarity is based on the derived fingerprints. The fuzzy hashing scheme is deterministic in that if two apps from two groups are identical, the same fingerprints will be generated. In addition, it can also effectively localize the changes possibly made in repackaged apps.[4]
Based on the above analysis, the similarity between the (shorter) fingerprints represents how similar their corresponding apps are. Similarity scoring algorithm computes the edit distance between these two fingerprints, which is the number of minimum edit operations, including insertion, deletion and substitution of a single byte, needed to convert one fingerprint into another. The algorithm Droid MOSS for Similarity Scoring is presented in Algorithm2. [10]

Algorithm 2:

```
Calculate the edit distance between two apps
Input: Two fingerprints fp1 and fp2
Output: Edit distance between fp1 and fp2
1: len1 strlen(fp1)
2: len2 strlen(fp2)
3: initialize_two_dimensional_matrix (matrix, len1, len2)
4: for i = 0 ! len1 do
5: for j = 0 ! len2 do
6: if fp1[i] = fp[j] then
7: cost = 0
8: else
9: cost = 1
10: end if
11: matrix[i, j] = min (matrix [i-1, j]+1, matrix[i, j-1]+1 matrix[i-1, j-1] + cost)
12: end for
13: end for
14: return matrix(len1, len2)
```

In particular, for two fingerprints fp1 and fp2 (with lengths of len1 and len2, respectively), reserve a two dimensional matrix (each value in the matrix is initialized to 0) to hold the edit distance between all prefixes of the first fingerprint and all prefixes of the second, and then compute the values in the matrix by flood filling the matrix. The distance between the two full strings will be the final value of the edit distance between the two fingerprints. The edit distance of any prefix subsequences of fp1 and fp2 can be derived from the minimum of three values: (1) matrix (i-1, j +1, which means to add one insertion operation in fp1; (2) matrix (i, j-1) +1, which means to add one deletion operation in fp2; and (3) matrix (i-1, j -1) + cost, which means to add one substitution operation between fp1 and fp2. Based on the calculated edit distance, we can derive a similarity score between two fingerprints. The formula used, is as follows:

$$\text{similarityScore}=\left[1-\text{distance}/\max\{\text{len1},\text{len2}\}\right]*100$$

If the calculated similarity score between two apps exceeds certain threshold and these two apps are signed with two different developer keys, the system reports the one not from the official Android Market as repackaged. The threshold selection affects both false positives and false negatives of our system. Specifically, a high threshold likely leads to low false positives but also high false negatives while a low threshold introduces high false positives but with low false negatives. Hence it is very important to determine the value of threshold to reduce the false positives and false negatives. Experiments reveal that threshold 70 is a good balance between these two metrics [10].

5. CONCLUSION:

Security of Mobile Computing is still a nascent field, with lots to research and a long way to go to achieve "complete security". The security of mobile computing presents new grounds for research as some of the problems faced in the mobile world are non-existent in the traditional wire-based computing environment. Future work could address information security related to the following three sub-areas of the mobile environment:

- The security of information residing in the mobile units, and the correctness and integrity of data in these mobile units.
- The security of information as it travels "over the air" between mobile units and mobile support sta-

tions. An important consideration in this area is the power consumption of the algorithms and schemes that implement this secure data transfer.

- New secure data storage schemes and data organization techniques will be required to facilitate rapid searching and transfer of data to and from mobile units.

REFERENCES

[1] Dustin Hurlbut, "Fuzzy Hashing for Digital Forensic Investigatior", Technical Report, Access Data Inc., May 17, 2011
[2] Sujithra.M, Padmavathi.G, "Mobile Device Security :A Survey on Mobile Device Threats, Vulnerabilities and their Defensive Mechanism", International Journal of Computer Applications, October 2012
[3] Threats and Security Issues in Mobile Computing SonikaȦ* and Sangeeta RaniȦ BLS Institute of Technology Management, Bahadurgarh-124507,NCR (Haryana), India Accepted 05 Oct 2014, Available online 10 Oct 2014,Vol.4, No.5 (Oct 2014) International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161
[4] Wu Zhou, Yajin Zhou, Xuxian Jiang, Peng Ning, "Detecting Repackaged Smartphone Applications in Third-Party Android Marketplaces"
[5] Yajin Zhou, Xuxian Jiang, "Dissecting Android Malware: Characterization and Evolution", 2012 IEEE Symposium on Security and Privacy.
[6] Forman, G.H. and Zahorian, J. (1994) "The Challenges of Mobile Computing". IEEE Computer, April 1994, 38-47.
[7] Survey on Mobile Computing Security 978-1-4799-2578-0/13 $31.00 © 2013 IEEE DOI 500 10.1109/EMS.2013.89
[8] Christoph Stach and Bernhard Mitschang, "Privacy Management for Mobile Platforms – A Review of Concepts and Approaches", 2013 IEEE 14th International Conference on Mobile Data Management.
[9] Characterizing the Performance of Security Functions in Mobile Computing Systems Abdulmonem M. Rashwan, Member, IEEE, Abd-Elhamid M. Taha, Senior Member, IEEE, and Hossam S. Hassanein, Senior Member, IEEE
[10] S. Bhargava and D. P. Agrawal, "Security Enhancements in AODV Protocol for Wireless Ad Hoc Networks," Proceedings of IEEE Vehicular Technology Conference, Atlantic City, 2001, pp. 2143-2147
[11] Heqing Huang, Sencun Zhu, Peng Liu, Dinghao Wu, "A Framework for Evaluating Mobile App Repackaging Detection Algorithms"
[12] Mavridis.I, Pangalos.G, "Security Issues in a Mobile Computing Paradigm", 2012
[13] Heqing Huang, Sencun Zhu, Peng Liu, Dinghao Wu, "A Framework for Evaluating Mobile App Repackaging Detection Algorithms"
[14] Security Guidance for Critical Areas of Mobile Computing, V1.0 at http://www.cloudsecurityalliance.org,
15] Lookout Mobile Security, "Lookout Mobile Threat Report", August 2011