Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2011) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

The 8th International Conference on Mobile Web Information Systems (MobiWIS)

Exploration of Attacks on Current Generation Smartphones

Steven Salerno, Ameya Sanzgiri [a1]*, Shambhu Upadhyaya

[a]*Department of Computer Science, University at Buffalo, Buffalo, NY-14260*

Abstract

Due to the ever increasing capabilities of current generation smartphones, they are quickly becoming more attractive targets for malicious attackers. The potential of porting attacks and malware from modern computers to these mobile devices is becoming a reality. In this paper, we explore the possibility of staging some attacks on the 802.11 network interface which is common to all smartphones. We begin by explaining and carrying out the exploitation of the SSH vulnerability on jailbroken iPhones that was discovered in late 2009. This paper then looks at simple network flooding attacks with the intention of causing a simple denial of service by depleting the battery life of the device. It is also our intention to show that these flooding attacks can be carried out utilizing a smartphone as the aggressor in order to attack other mobile devices and that the procedure for such attacks is not difficult. A simple tool is developed in order to carry out these attacks and to show that even though these attacks are relatively simple, they can have profound effects.

Keywords: Smartphones; Denial of Service; Network Security

1. Introduction

Today mobile devices have become ubiquitous due to their ability to provide a wide variety of services to users. Current generation mobile hardware and operating systems, such as Apple iOS [1], Google Android [2, 3], Windows Mobile 7 [4], and Blackberry RIM [5], have begun to rival personal computers in performance as well as capabilities. However as mobile devices and their operating systems continue to grow and evolve, security becomes a major concern due to the possibilities of these devices being threatened and exploited in manners that are similar to modern computers.

The fact that these devices are so common makes them easy potential targets for an attacker and could be thought of as an attack platform similar to any other computer. Mobile devices often times store personal data, such as contact details, SMS messages, etc., as well as utilize a variety of applications that can prompt for and store additional information. This information can be valuable and potentially damaging if an attacker can retrieve it from the mobile device that it is stored on. Mobile devices would also make attractive aggressors since by default they are small in size, making them easy to conceal, highly portable, and are yet to be truly characterized as devices to launch attacks from.

In this paper, we look at both the vulnerabilities of these devices as well as the attack capabilities of the current generation smartphones through a Wi-Fi connection. First we present the Secure Shell (SSH) vulnerability that

* Corresponding author. Tel.: 716-548-8126;
E-mail address: ams76@buffalo.edu.

exists on jailbroken (see Sec. 3.1) iOS devices and investigate if a similar exploit exists within the Android operating system. We then investigate the possibility of using these devices in Denial of Service (DoS) attacks through SYN ACK and ping flooding. We present the details of a tool that we developed, which is capable of finding devices on a wireless network and carrying out these attacks. Finally, we look at the case where computers can be used as attack devices to exploit vulnerabilities of these devices. To the best of our knowledge this line of research is the first to investigate the use of smartphones as tools to attack networks.

We enumerate our contributions as follows:

1. Investigating the SSH vulnerability that affects iOS devices
2. Studying the feasibility and effects of DoS attacks against smartphones
3. Investigating the scenario of smartphones as attack tools

The paper is organized as follows – Section 2 outlines the work in the literature related to smartphone security. Section 3 describes the iOS SSH vulnerability, how it can be exploited and an analysis of the same vulnerability in Android. Section 4 provides an overview of the attacks we consider in this work, the methodology of the attacks and the attack scenarios. Section 5 presents the details of the tool we create for this research and the attack steps involved. Section 6 presents the results of the experiments. Finally, Section 7 concludes the paper and provides overall thoughts.

2. Background and Related Work

Until recently, research on smartphones has been limited to investigating ways to utilize these devices for processing or using phone sensors for sensing applications [6, 7]. However, with the increase in smartphone's capabilities, the area of smartphone security has also gained importance. Research has been done into studying the vulnerability of smartphones to rootkits and malware [8-10].

The authors of [11] describe how mobile phones can be intended targets of network attacks and provide a methodology for protecting these devices against TCP flooding attacks. However, their work considered the attacks over a cellular network. Recently there also has been work relating to privacy and confidentiality [12].

While all the research up until now has looked at various aspects of security, no one has really studied the effects of network attacks against smartphones. Given the features and the popularity of current generation smartphones we propose that one needs to investigate the effects of smartphones to different attacks. However, studying the effect of the attacks on smartphones is just one side of the coin. We also believe that given the small form factor, smartphones are likely to make attractive attack tools and it is imperative to investigate a scenario that uses smartphones as aggressors.

3. Exploiting SSH on Mobile Devices

One of the capabilities that users may add to a mobile device is the installation of a SSH server. This allows the user to remotely access their device without having to physically tether it to a computer. While this practice may greatly increase the ease of accessing files it requires that device be jailbroken. The jailbreaking or rooting process (depending on the device's operating system) allows for the installation of unofficial and/or unauthorized applications on the device. It is important to note that jailbreaking and rooting are frowned upon by both manufacturers and mobile network carriers, and typically voids the warranty.

3.1. iOS JAILBREAKING AND SSH

In order to gain the ability to remotely access an iOS device through SSH, the device must first be jailbroken. Jailbreaking allows the operating system to gain root access and install applications that have not been approved by Apple's Certificate Authority. The process generally exploits some security flaw that is present on the mobile device. Early generation iOS devices could be unlocked by exploiting the *libtiff* vulnerability, which is further detailed in [13]. Current generation jailbreaking relies on exploiting vulnerabilities present within the bootrom which typically needs a hardware revision to patch.

Usually the process of discovering and exploiting these vulnerabilities is generally difficult for the end users as such. However, the process has been made relatively simple for end users by groups which have developed some tools for the same. Some examples of popular jailbreaking tools are the iPhone Dev Team's redsn0w, PwnageTool [14], Greenpois0n by the Chronic Dev Team [15] and Sn0wbreeze by ih8sn0w [16]. While jailbreaking an iDevice (devices that run on iOS) users are given the option to install Cydia [17] an open source alternative to Apple's App Store. Cydia's installation comes packaged with an SSH server, but users also have the option to install the OpenSSH application through Cydia.

3.2. iOS SSH VULNERABILITIES

Jailbreaking an iOS device opens a potential vulnerability that can be exploited especially when an SSH server is actively running. Since the early versions of iOS, the superuser and password are assigned the default values "root" and "alpine" respectively, and with this knowledge anyone can connect to an SSH enabled iDevice as a root user. Using both the root access and file transfer capabilities of SSH, an attacker could steal any amount of information from the device, push files such as root kits, malware, etc., on the device, or execute any system call.

To exploit this, attackers need only scan a wireless network, find iOS devices running SSH and attempt to connect using the default username and password. This process is fairly simple using any network. While this vulnerability is fairly simple, it exists because of two main reasons: (1) the Jailbreaking process makes it very easy for even novices to install third party applications, and (2) many users who jailbreak their iOS devices often times do not know how to or care to change their superuser password or take other preventative measures that are available to them.

3.3. REMEDIES AGAINST iOS SSH VULNERABILITY

There are several methods that users with jailbroken devices can use to prevent attackers from exploiting the above vulnerability. The first and most simple remedy is to change the root password, which can be done through the passwd utility or by directly modifying the master.passwd file. Of course this method is still subject to dictionary attacks if weak passwords are used. It is also possible to set up RSA key authentication over SSH instead of the use of passwords. This of course involves additional steps such as the generation of the keys and their proper placement on devices (which may prevent some users from attempting this process). It would also be helpful to kill the SSH daemon when its utilities are not needed by the user. This would greatly minimize the risk from having the service constantly running.

3.4. ANDROID ROOTING AND SSH

Unlike iDevices, the Android based devices do not need root access to install and use an SSH server. This is because developers of applications can publish them after digitally signing them due to the lack of a rigorous certification process. There are several applications available on the Android Market, such as QuickSSHd [18], that allows a user to run an SSH server without the need of rooting the device.

While it is not necessary, one still can root an android device by exploiting the firmware and/or hardware. This makes the rooting process slightly more difficult because manufacturers often customize the Android operating system to suit their specific hardware platforms. Due to this all the "flavors" of Android may not have the same vulnerabilities. Also manufacturers have begun to release hardware with security features to prevent rooting, often times causing the device to fully fail if the process is attempted.

One example of one such tool used to root several HTC mobile devices is unrevoked [19]; it utilizes an exploit found by Sebastian Krahmer. Once the device is rooted a manual installation of an SSH server, such as Dropbear is possible. Unlike the iDevices, the SSH daemon will begin on restarting the device.

3.5. ANDROID SSH VULNERABILITIES

Unlike iDevices, those that are running the Android operating system that is rooted and have an SSH server, do not have a similar vulnerability. This is due mainly because of the fact that it has been deemed that – either no superuser password exists or can be found. Hence, there is no default set of values for an attacker to use. Furthermore, in the case that the user manually installs Dropbear, he has to specify the superuser name and password values. For the more advanced users, Dropbear can be configured to use RSA key authentication making the vulnerability much more difficult to exploit.

4. Mobile Devices in a Network

While current generation smartphones have continued to grow more powerful in terms of both hardware capabilities and processing power than their predecessors they still have one major constraint, namely–power consumption. As a result it may be attractive for an attacker to simply kill off the device, using, say, a denial of sleep attack, rather than attempting to steal information from it. By utilizing network attacks, an attacker can consume the battery life of the device by causing the radio to be constantly receiving and possibly transmitting. We focus on the flooding attacks, SYN and ping, in order to generate a great deal of network traffic focused solely on the intended mobile device.

4.1. ATTACK OVERVIEW

4.1.1. SYN ATTACK

For our purposes we do not implement a SYN flood as it is referred to in [20, 21], but use a variant of it to generate a large amount of network traffic in a short time frame. In this variant of attack, one opens and closes TCP connections to the target device repeatedly and as quickly as possible. This will, instead of impairing the TCP service, generate a series of SYN, ACK packets in accordance with the three way handshake required at the start of every TCP connection. By immediately closing the connection once the handshake has finished, one can then restart the handshake by opening another connection. The underlying assumption while carrying out this method to generate traffic is that the mobile device has some TCP ports open for connections. Without open TCP ports there would be no active services running on the device to connect to and hence no network traffic would be generated. We implemented this kind of an attack to target mobile devices primarily to leverage the power constrained smartphones. However, such an attack could also be easily carried out against other devices such as laptops.

4.1.2. PING FLOODING ATTACK

Ping flooding [22] is an attack based on a built-in feature namely the ping command on most operating systems. However, only the Unix operating system supports the '`–f`' flag. When used with this flag, the ping command will send packets as fast as can be supported by the network or up to one hundred times per second, whichever the system deems is greater. The ping command sends an ICMP ECHO_REQUEST packet and waits for an ICMP ECHO_RESPONSE to be received from the host. By executing this command against a device one can generate a massive amount of network traffic targeting only the device. Ping flooding does not have the same downfall as the above attack since it does not require any open TCP ports to generate traffic.

4.2. SYNTHESIZING OUR ATTACK SCENARIOS

In this paper we are concerned with two different scenarios, the first being more powerful devices, such as laptops that can have a constant power (if required) attacking smartphones. This scenario conforms with a typical model, as we often expect the more powerful devices to act as an aggressor. As mentioned above, in this scenario we are concerned with depleting the smartphone's power since as outlined in Section 3 stealing information from the smartphone requires the device to be jailbroken, a situation we believe might not be very common.

The second scenario we investigate is the smartphone acting as the aggressor and attacking its more powerful counterpart such as a laptop. While traditionally one would not have the same expectations from this model as we do with the first, it is important to note that smartphones do have some advantages. First, as they are not perceived to act as aggressors, they are likely to be less sought after when an attack is being carried out. This means they are more attractive to attacker's as they can be concealed by an attacker. Further, since smartphones are resource constrained, many of their capabilities are optimized to be more efficient (in terms of memory and power consumption) than their powerful counterparts. However, the lack of additional features in terms of both hardware (such as USB ports) and software (firmware) also limits these devices. For example, smartphones cannot be used to sniff packets over the wireless medium unlike a laptop where an attacker can add network cards (via a USB dongle) and sniff packets in promiscuous mode.

4.3. SYSTEM MODEL OF ATTACKS

All experiments were conducted in a controlled environment with only our devices connected to the network. This was done primarily to check the effects of the attack as well as the security policies of the university where this research was conducted. The smartphone used for our experiments was HTC's Droid incredible [23]. The Droid Incredible by HTC comes with Android 2.2 and is equipped with a 1 GHz Qualcomm Snapdragon processor. The smartphone also comes with 512Mb RAM and is Wi-Fi®: IEEE 802.11 b/g capable. Two laptop computers – one with the Ubuntu OS and one with the Windows OS were used as the powerful counterparts to the smartphones. The Laptop with the Windows OS was used for the SYN ACK flood attack and ran a tool we created for that purpose. The Ubuntu laptop was used for the ping flood attack (since the Unix system supported the `-f` flag we needed). The Ubuntu laptop was also the target for the attacks initiated from the smartphone. For the ping flood attack, the Windows system was used to observe the network speeds as we envisioned some possible side effects of the ping flood.

Before we proceed to the results of our experiments, we first present details of the tool created to automate the SYN ACK flood attack and also the attack procedures in the following section.

5. SYN ACK Attack Tool and Attack Procedures

In order to automate the process needed for discovering and exploiting mobile devices connected to a wireless access point we developed a tool. The tool was implemented in Java and provides the user a relatively straightforward interface to carry out the exploits. The backend of the tool relies heavily on the popular network exploration tool Nmap [24].

Using the tool one can perform an initial host discovery on a wireless network and determine all 'online' hosts in a user specified subnet. It is important to note that this required the device on which the tool will be used to be connected to the same wireless network. One can then use the command to perform the initial host discovery scan. An example of the command would be as follows:- `nmap –sP 192.168.1.100/24 –exclude 192.168.1.100.`

The details of the commands are as follows: the `–sP` flag specifies to the tool to use Ping Scan, a common utility in Nmap. The next argument namely, `192.168.1.100/24`, specifies the subnet of the scan which by default is 24 but can be specified by the user. Thus, if the IP address `192.168.1.100` is that of the computer the tool is currently running on, the ping scan would check the IP addresses from 192.168.1.100 to 192.168.1.256. The exclude is added to eliminate the host computer from the scan.

This preliminary scan allows us to get the IP address, MAC address and NIC vendor (which is derived from the MAC address) for each host discovered on the network. Once a list of active hosts is obtained, a more intense scan can then be performed to find the open TCP ports as well as the device's operating system. The following Nmap command, `nmap –O –p1-65535 ip_address`, performs the OS detection scan on `ip_address` and scans all TCP ports. This additional information is crucial when we are deciding which targets to attack.

Once any open TCP ports on the host are discovered, one can start the SYN ACK attack and flood the device. Thus, the steps for performing a SYN ACK flood against a smartphone are as follows:

1. *Start the tool.* A host discovery of the default subnet is performed based on the user's IP address. The hosts that are found are displayed in a list format on the left side of the tool. The user can also select a new subnet and perform the host discovery again at any time.
2. *Scan a host.* After selecting a host in the list and pressing the Scan button, an OS detection and scan of all ports takes place and the results are returned to the user in the right window.
3. *Flood a host.* Once a host has been scanned, if any TCP ports are in an open state the tool can begin to SYN ACK flood by selecting the Flood button, however while the attack is running the tool cannot perform any other operations. The attack can be cancelled at any time.

It is important to note that the SYN ACK flood attack is an attractive option for an attack, since most smartphones lack a firewall and the effects of the attack are undetectable from the user's point of view. However as explained before it requires the phone to have some open ports.

The ping flood is not implemented on the tool since the ping flood capability is present on Linux and Unix. To carry out this attack one can simply open a command shell on a Linux machine and enter the following command, `ping –f ip_address`, where `ip_address` is the host that we wish to flood. The tool as described above can be used to find the IP address. This technique of course does not depend on open ports and could be carried out after the preliminary host discovery scan.

6. Experiments and Results

We use the measurements from our model as explained in Sec. 4.3 to evaluate the effectiveness of the attacks in our attack scenarios. All experimental results presented here depict the average values over three runs.

6.1. LAPTOP AS AGGRESSOR

6.1.1. ATTACK METHODOLOGY

As our attacks aimed at depleting the power of the smartphones, we needed to perform experiments in two scenarios. The first scenario was to establish a baseline for the battery life of the smartphone and was determined in the absence of attacks. Next by performing the attacks on the smartphone we determined the effectiveness of the attacks. During all of these tests the device was turned on, started at full battery life, the screen was kept on at full brightness and the volume was kept at 50%.

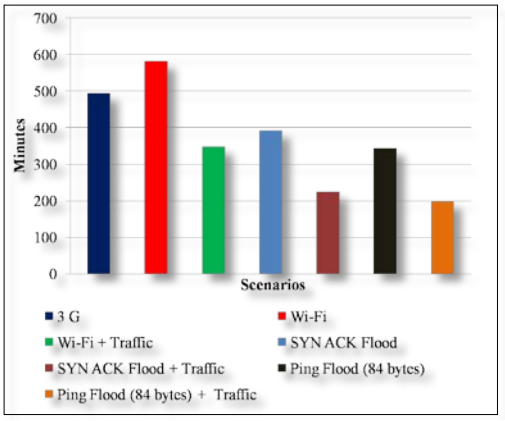

Figure 2. Smartphone battery consumtion under various scenarios

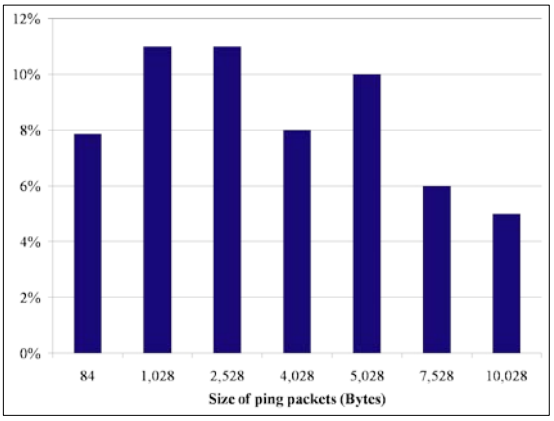

Figure 1. Effect of the size of Ping Packets on battery life when ping flooded for 20 minutes

While determining the baseline for battery life we needed to understand the power consumption of individual components and various network interfaces, to model the power consumption of smartphones in a typical day to day real-world scenario. Thus, we first determined the baseline battery life with only the 3G interface of the smartphone

on. Next only the Wi-Fi interface was switched on but no network traffic was generated. Finally we generated web traffic by accessing videos on YouTube using the Wi-Fi interface in order to simulate a more realistic scenario.

To determine the effectiveness of attacks by themselves we performed the SYN ACK and ping flooding on the targeted smartphone without any additional traffic. Finally we performed the SYN ACK flooding and ping flooding while generating web traffic to determine how effective these attacks are on a smartphone.

6.1.2. RESULTS

Figure 1 shows the smartphone's battery consumption under each of the scenarios discussed above. It is clear that both of the attacks are highly effective in draining the battery, however, the ping attack is especially more effective and attractive to an attacker. This is due to the fact that the ping flood requirement is less strict, unlike the SYN ACK flood that requires TCP ports to be open. Further, the ping flood also had a side effect on the network as it DoS-ed the network and network speeds deteriorated drastically. For thoroughness and to further study the effect of ping packet size (which can be user defined) we conducted the ping flood attack for 20 minutes varying the ping packet sizes.

Figure 2 illustrates the results of the above experiments. As can be seen, the percentage of battery consumption is greatest when the ping size packets are 1028 bytes and 2028 bytes. A further increase in the ping packet size does not result in an increase in the battery power consumption.

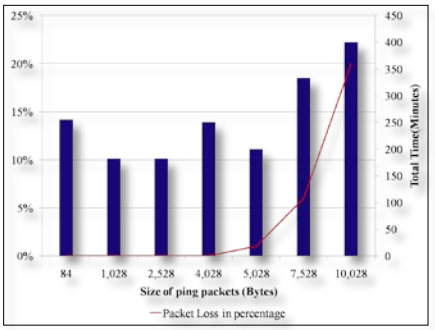

Figure 3. Battery Drain Time for various ping packet sizes. Packet Loss percentages observed for corresponding sizes

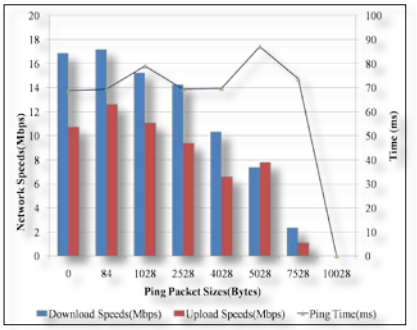

Figure 4. Effect of various ping packet sizes on Network Characteristics

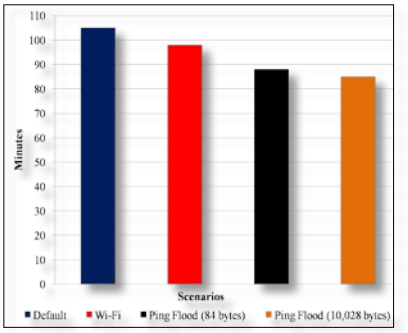

Figure 5. Laptop Power Consumption under various scenarios

Figure 3 illustrates the results of our experiments and corroborates our observations. Figure 4 depicts the severity of ping flood on the network as the size of ping packets increases. While these experiments show that the minimum time taken to completely drain the battery of a smartphone is a minimum of 3 hours, a ping flood attack would deny service to all clients connected to the network very quickly.

6.2. SMARTPHONE AS AGGRESSOR

6.2.1. ATTACK METHODOLOGY

Based on our observations from the above experiments we decided to perform only the ping flood attack. The attack can be performed as similar to the above case since one can access the command line prompt in Android with several applications.

6.2.2. RESULTS

Figure 5 illustrates the results of the ping flood attack on the laptop using a smartphone. It is interesting to note that the ping flood is not as effective on the laptop as it was on the smartphone, since unlike in smartphones where the interfaces transition into "sleep" mode when not used, the laptop interfaces are always on. The DoS effect on the network remains the same.

7. Discussions and Conclusion

In this paper we investigated the security of smartphones and the SSH vulnerability of iDevices. We demonstrated how Android devices do not suffer from the same vulnerability. We also presented two novel attack scenarios involving smartphones. We demonstrated that while smartphones do present an attractive target to launch DoS attacks against networks targeting the power consumption of smartphones is infeasible. Similarly we also demonstrated that while envisioning of the SYN ACK attack is attractive due to the lack of firewalls on smartphones, it requires too many conditions to be satisfied for success and is thus infeasible. However, given the battery capacity and form factor of a smartphone, they make an attractive option as an attack tool.

While the attacks we have presented show the capabilities of smartphones as both targets and aggressors, they are a small subset of the many network attacks that are possible. Our next step will be to investigate other attacks for deeper insights into smartphone based attacks.

Acknowledgments

This research has been supported in part by National Science Foundation Grant DUE-0830814. The usual disclaimers apply.

References

1. Apple. Apple iOS Software. Available from: http://www.apple.com/ios/.
2. Developers, A. Android and Developement. Available from: http://developer.android.com/sdk/index.html.
3. Enck, W., M. Ongtang, and P. McDaniel, Understanding Android Security. Security & Privacy, IEEE, 2009. 7(1): p. 50-57.
4. Microsoft. Microsoft Windows Mobile 7. Available from: http://www.microsoft.com/windowsphone/.
5. Motion, R.I. Research in Motion- Blackberry. Available from: http://www.rim.com/.
6. Beurer-Zuellig, B. and M. Meckel. Smartphones Enabling Mobile Collaboration. in Hawaii International Conference on System Sciences, Proceedings of the 41st Annual. 2008.
7. Sian Lun, L. and K. David. Movement recognition using the accelerometer in smartphones. in Future Network and Mobile Summit, 2010. 2010.
8. Dixon, B. and S. Mishra. On rootkit and malware detection in smartphones. in Dependable Systems and Networks Workshops (DSN-W), 2010 International Conference on. 2010.
9. Mulliner, C. Vulnerability Analysis and Attacks on NFC-Enabled Mobile Phones. in Availability, Reliability and Security, 2009. ARES '09. International Conference on. 2009.
10. Cong, J., H. Xiaoyan, and J. Songlin. Propagation Model of Mobile Phone Virus Based on Efficiency of Immunization. in MultiMedia and Information Technology, 2008. MMIT '08. International Conference on. 2008.
11. Swami, Y.P. and H. Tschofenig. Protecting mobile devices from TCP flooding attacks. in Proceedings of first ACM/IEEE international workshop on Mobility in the evolving internet architecture. 2006.
12. Loukas, A., et al., MILC: A secure and privacy-preserving mobile instant locator with chatting. Information Systems Frontiers, 2010. 12.
13. Pandya, V. and M. Stamp, iPhone Security Analysis. Journal of Information Security, 2010. Volume 1(No. 2): p. 74-87.
14. BLOG, D.T. DEV-TEAM BLOG. 2011; Available from: http://blog.iphone-dev.org/.
15. MEMBERS, C.D. Chronic Dev Blog. 2011; Available from: http://chronic-dev.org/blog/.
16. ih8sn0w.com. ih8sn0w.com | Jailbreak your iPod Touches and iPhones. Available from: http://ih8sn0w.com/index.php/welcome.snow.
17. Freeman, J. Cydia. 2010; Available from: http://cydia.saurik.com/.
18. TeslaCoil. QuickSSHd | TeslaCoil Software. Available from: http://teslacoilsw.com/quicksshd.
19. unrevoked. Unrevoked - Set your phone free. Available from: http://unrevoked.com/.
20. Nashat, D., J. Xiaohong, and S. Horiguchi. Detecting SYN Flooding Agents under Any Type of IP Spoofing. in IEEE International Conference on e-Business Engineering, 2008. ICEBE '08. . 2008.
21. Nakashima, T. and S. Oshima. A Detective Method for SYN Flood Attacks. in Innovative Computing, Information and Control, 2006. ICICIC '06. First International Conference on. 2006.
22. Cabrera, J.B.D., et al. Proactive detection of distributed denial of service attacks using MIB traffic variables-a feasibility study. in Integrated Network Management Proceedings, 2001 IEEE/IFIP International Symposium on. 2001.
23. HTC, V. HTC mobile Phones. 2011; Available from: http://www.htc.com/us/products/droid-incredible-verizon#tech-specs.
24. NMAP. NMAP- Free Security Scanner for Network Exploration & Security Audits. Available from: http://nmap.org/dist/nmap-5.51-win32.zip.