IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 14, Issue 3 (Sep. - Oct. 2013), PP 104-109
www.iosrjournals.org

Penetration Testing for Android Smartphones

Okolie C.C[1], Oladeji F.A[2], Benjamin B.C[3], Alakiri H.A[4], Olisa O.[5]
[1]*(Computer Sciences, University of Lagos, Nigeria)*
[2]*(Computer Sciences, University of Lagos, Nigeria)*
[3]*(Computer Sciences, University of Jos, Nigeria)*
[4]*(Computer Sciences, Yaba College of Technology, Nigeria)*
[5]*(Computer Sciences, University of Lagos, Nigeria)*

Abstract: *One major challenge faced by Android users today is the security of the operating system especially during setup. The use of smartphones for communication, social networking, mobile banking and payment systems have all tripled and many have depended on it for their daily transactions.AndroidOS on smartphones is so popular today that it has beaten the most popular mobile operating systems, like RIM, iOS, Windows Mobile and even Symbian, which ruled the mobile market for more than a decade.This paper performspenetration testing of Android-based Smartphones using an application program designed to simplify port-scanning techniques for information gathering and vulnerability attack. In this paper, an attempt was made to test and analyze the security architecture of the Android operating system using the latest penetration testing and vulnerability tool based on Kali Linux. Three different Versions of Android, Version 2.3, 3.2 and 4.2 were simulated on a virtual machine.*

The result shows that although there is an improvement in the security stack of the different AndroidVersions but Version 4.2 is more secured than the others. This work is important for users and researchers who use their Android smartphones in a critical environment with hostile network traffic.

Keywords: *Android, Linux, Penetration testing, Security, Smartphones*

I. Introduction

Research has shown that smartphone usage within the last few years has increased rapidly and this is due to their rich and versatile functionality. Today, Smartphones are not just used as a communication tool; but for other functionality like Pager, PDA (Personal Digital Assistants), MID(Mobile Internet Devices), GPS, MP3 Player [1] etc., and provides a range of services like social networking, entertainment, electronic banking, reading e-books or online meetings. These services and functionality is possible due to the use of astrongstable operating system Androidthat is very fast and reliable [1]

The major problem is the way these services are introduced and released by smartphone vendors. Security implications of these devices and the network on which smartphones are used should be of much concern. In most cases, hackers take advantage of the porous nature of these services and applications to exploit the smartphone and render it unusable by installing applications that will make it to freeze, run out of memory, or spoof communication with other devices.

The rapid growth and increase in the use of Android Smartphones has exposed it to a lot of hacking activity by hackers trying to exploit and attack the smartphone especially as its operating system is based on an open platform. Penetration testing is a very good method for detecting the vulnerabilities of a system; this is because it helps in finding security holes in the system. A penetration test is a method of evaluating the security of a computer system or network by simulating an attack from a malicious source. The methodology for how to perform penetration tests is given by the National Institute of Standards and Technology [2,3]. Most studies regarding security of smart phones have mainly focused on the application layer, such as viruses, worms, MMS exploitation and Cross-Service Attacks [4]. A similar penetration test on Android smartphone is one carried out in 2011 [5] where an earlier Version of Android was tested and analyzed for vulnerability.

In this paper, an application program was developed for scanning and exploitation on the smartphones. This application program is used to simplify the discovery and information gathering stage. From the information gathered, a vulnerability test was conducted using a penetration-testing tool called Kali Linux. This tool was used to perform a white hat penetration test and exploit any vulnerable ports on the smartphones.

This paper is organized as follows: Section one introduces the issue concerned, section two describes the overview of Android operating system architecture, section three describes the penetration test simulationtopology while section four discusses the summary of result and findings. The conclusion was drawn in section five.

II. Related Works

G1-Android is the first commercially available phone that features Google's Android Platform SDK. This is a partnership project with Taiwan Based HTC Corp and supported by the United States Service provider T-Mobile. Thus the phone is available in the market as T-Mobile HTC Android. Android is the first truly open source platform for mobile devices with a fully integrated software stack that consists of an operating system, middleware, user-friendly interface and applications, and also allows the users to develop additional software and change or replace functionality without limitations. In order to achieve the unlimited functionalities, Android uses Linux operating system as its core OS and ensures that users experience same Internet activities equal to what they can experience on a desktop PC [6, 7, 8, 9, 10]. Several smartphone features and functions help to increase usage of data and services but it is also open to the risk of introducing new vulnerabilities.

Android consists of a kernel based on Linux kernelVersion 2.6 with middleware, libraries and APIs written in C, and application software running on an application framework which includes Java-compatible libraries based on Apache Harmony [11]. Since Android operating system release in 2008, there are so many versions and each new release comes with new features and bug fixes

2.1Android Architecture

The Android operating system is referred to as a software stack of different layers, where each layer is a group of several program components. Together it includes operating system, middleware and important applications. Each layer in the architecture provides different services to the layer just above it [12].

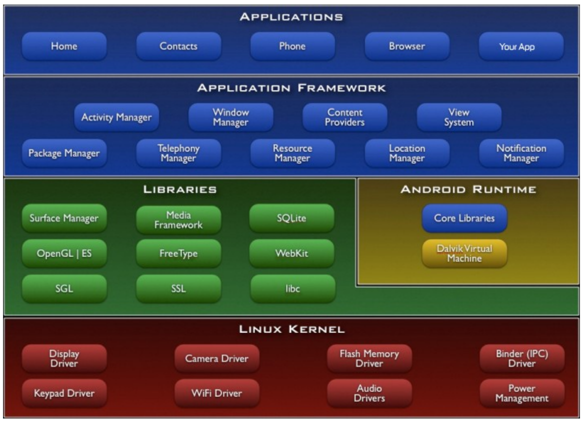


Fig. 1: diagram of Android architecture [13]

The following layers are of importance to this paper:

2.1.1 Linux Kernel

The whole Android OS is built on top of the Linux 2.6 Kernel with some further architectural changes made by Google. It is this Linux that interacts with the hardware and contains all the essential hardware drivers. Android uses the Linux for all its core functionality such as Memory management, process management, networking, security settings etc. As the Android is built on a most popular and proven foundation, it made the porting of Android to variety of hardware, a relatively painless task [13]

2.1.2Libraries

The libraries layer is the layer that enables the device to handle different types of data. These libraries are written in C or C++ language and are specific for a particular hardware.

2.1.3 Android Run Time

This layer consists of Dalvik Virtual machine and Core Java libraries, Dalvik VM is a type of JVM used in Android devices to run apps and is optimized for low processing power and low memory environments. It allows multiple instance of Virtual machine to be created simultaneously providing security, isolation, memory management and threading support

2.1.4 Core Java libraries

These libraries provide most of the functionalities defined in the Java SE libraries.

2.1.5 Application Framework

These are the blocks that our application directly interacts with. These programs manage the basic functions of phone like resource management, voice call management etc.

2.1.6Application

This is the top layer in the Android architecture and this is where applications run. Several standard applications come pre-installed with every device, such as SMS client app, Dialer, Web browser, Contact manageretc.

Most studies regarding security of smart phones have mainly focused on the application layer, such as viruses, worms, MMS exploitation and Cross-Service Attacks.Recent work has shown that there different ways of carrying out penetration test on smartphones. The most recent is one done by Naresh and Muhammed on Android based smartphones [14]. In their test, the authors looked at threeearlier Versions of Android releases. The only problem with the work is that AndroidVersions have tripled with a lot of bug fixes and modification in the security architecture of the recent Versions. So the test can only be true for earlier Version of the Android-based smartphones. This paper is an improvement in the earlier work using more simplified and sophisticated method for information gatheringand port scanning. The test was performed on recent release of the Android operating systems (Version 2.3, 3.2 and 4.2).

This paperfocused on the Android operating system deployed on a Virtual machine with respect to its core architecture and TCP/IP network stack security issues. The security analysis was performed using a white box penetration testing. Information on the target system were gathered and identified and then penetration tests performed, starting from well-known vulnerabilities to more specific deep penetration attacks. The test was performed on a simulated virtual machine with Android images using open source penetration tools. The simulation were carried on three different Versions of the Android operating system, Version 2.2 (Froyo), Version 3.2 (Honeycomb) and Versions 4.2.x (Jelly Bean) released in 2010, 2011 and 2012 respectively

III. Simulation Architecture

The architecture of the system used in this work for the penetration test on the Android smartphones is a simulation on a virtual machine. The test included port scanning, and running exploit tools in order to test the different layers of the TCP/IP network stack of the different AndroidVersions selected for the test. To simplify the test, an application program written in Objective C was used to manage the port scanning and exploit stage while the vulnerability found was exploited using an opensource penetration tool based on Kali Linux. Fig. 2 below shows the system architecture of the different AndroidVersions and the penetration-testing tool.

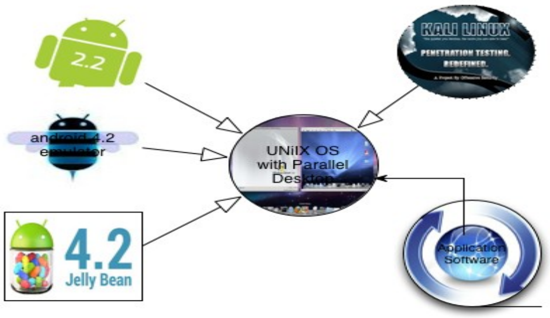


Fig. 2:architecture of the penetration test system

This report followed the methodology used in [14] with the simulation activities shown in Fig. 3 below:

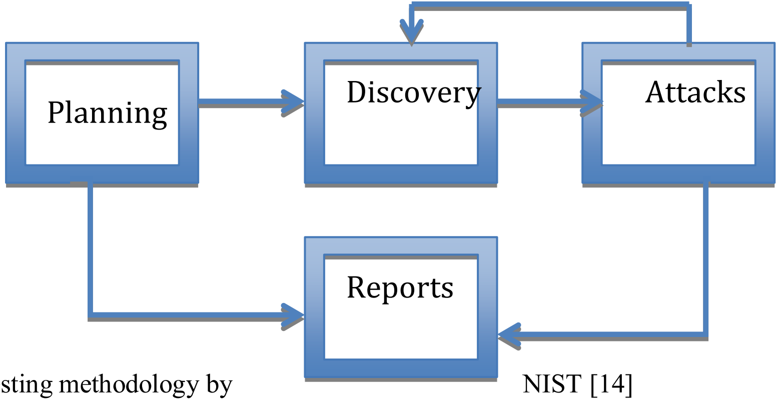


Fig. 3: penetration-testing methodology by NIST [14]

The planning phase is the setting up of the systems, installation and configuration of the simulation environment. In this phase, the different AndroidVersions was loaded and activated. The discovery phase is the information gathering phase in which the target system was scanned and open ports identified as well as active and inactive systems. The attack phase is the actual vulnerability test carried out on the target system. All the ports discovered as vulnerable are used to exploit the target system. Information gathered from the simulation was reported which is the final phase of the penetration test. The functionalities of the system simulation is shown in Fig. 4

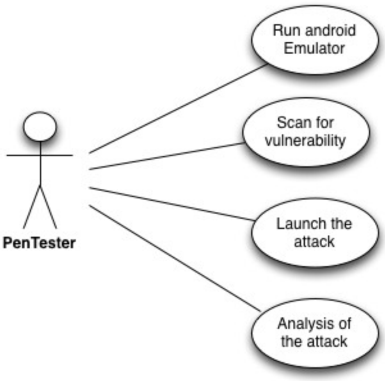
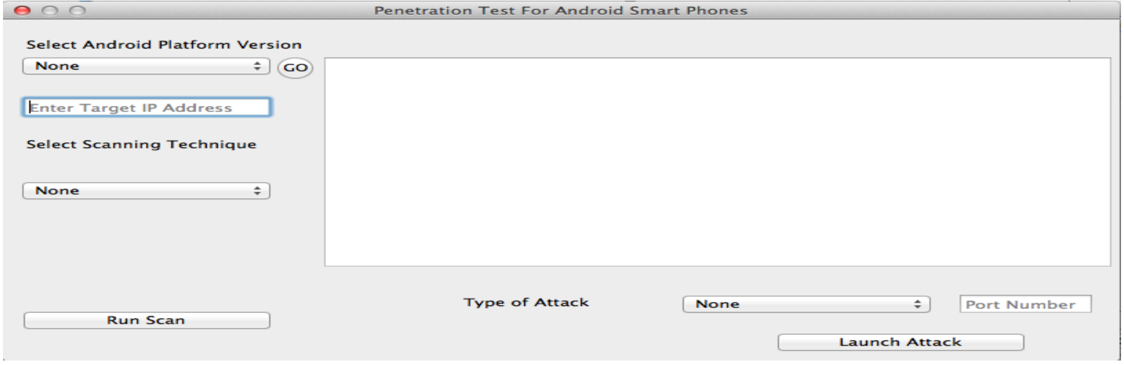


Fig. 4: use-case diagram of the proposed system

In Fig. 4 above, the pen tester launches the software application and selects the Version of Android to perform the exploit on. Once the system is scanned and open ports detected, an attack is launched and an analysis on the attacked ports is performed and reported.

IV. Discussion On Findings

The first stage in the implementation is the launching of the application software as shown in Fig. 5


Penetration Test For Android Smart Phones

Select Android Platform Version: None

Enter Target IP Address

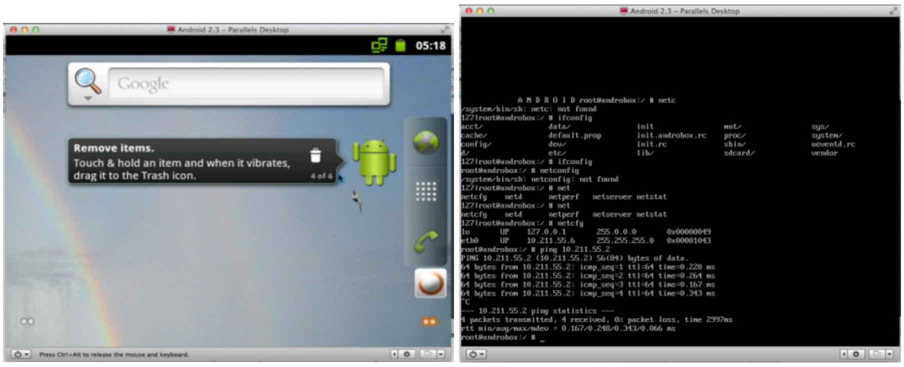
Select Scanning Technique: None

Type of Attack: None

Port Number

Fig. 5: interface of the penetration testing software.

From the Fig 5 above, the user selects the AndroidVersion to load and clicks on the GO button. The Android image loads up in two views as shown in Fig. 6 below. The first window displays the graphic user interface while the second window shows the CLI interface, which displays the network address assigned to the Android image

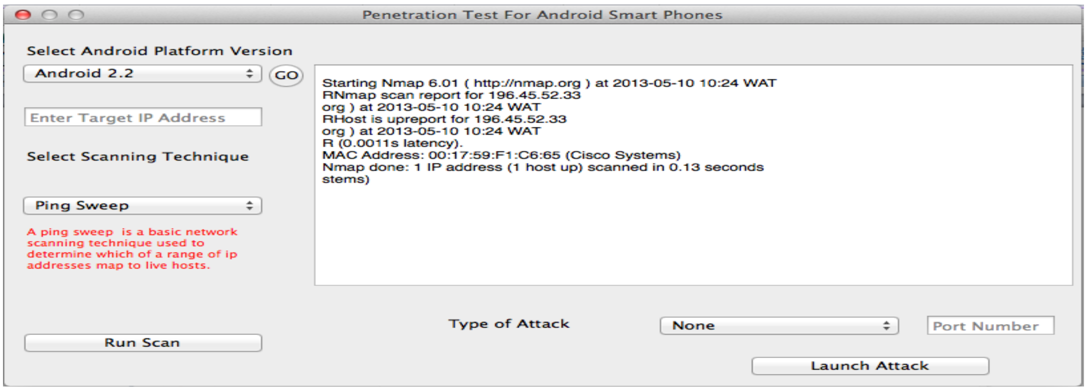

Android 2.3 – Parallels Desktop
05:18
Google
4 of 6
Press Ctrl+Alt to release the mouse and keyboard.
Android 2.3 – Parallels Desktop

Remove items.
Touch & hold an item and when it vibrates, drag it to the Trash icon.

```
            A N D R O I D root@androbox:/ # netc
/system/bin/sh: netc: not found
127|root@androbox:/ # ifconfig
acct/           data/           init            mnt/            sys/
cache/          default.prop    init.androbox.rc proc/          system/
config/         dev/            init.rc         sbin/           ueventd.rc
d/              etc/            lib/            sdcard/         vendor
127|root@androbox:/ # ifconfig
root@androbox:/ # netconfig
/system/bin/sh: netconfig: not found
127|root@androbox:/ # net
netcfg    netd      netperf   netserver netstat
127|root@androbox:/ # net
netcfg    netd      netperf   netserver netstat
127|root@androbox:/ # netcfg
lo       UP    127.0.0.1       255.0.0.0       0x00000049
eth0     UP    10.211.55.6     255.255.255.0   0x00001043
root@androbox:/ # ping 10.211.55.2
PING 10.211.55.2 (10.211.55.2) 56(84) bytes of data.
64 bytes from 10.211.55.2: icmp_seq=1 ttl=64 time=0.220 ms
64 bytes from 10.211.55.2: icmp_seq=2 ttl=64 time=0.264 ms
64 bytes from 10.211.55.2: icmp_seq=3 ttl=64 time=0.167 ms
64 bytes from 10.211.55.2: icmp_seq=4 ttl=64 time=0.343 ms
^C
--- 10.211.55.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.167/0.248/0.343/0.066 ms
root@androbox:/ # _
```

Fig. 6:gui and cli interface of the Android image

The next phase is entering the IP address of the target host, which is the Android image selected and the type of scan to perform will be selected. Then the Run Scan button will be activated as shown in Fig. 7. This will then query the target host and gather information from it.


Penetration Test For Android Smart Phones
Select Android Platform Version
Android 2.2
GO
Enter Target IP Address
Select Scanning Technique
Ping Sweep
Run Scan
Type of Attack
None
Port Number
Launch Attack

A ping sweep is a basic network scanning technique used to determine which of a range of ip addresses map to live hosts.

```
Starting Nmap 6.01 ( http://nmap.org ) at 2013-05-10 10:24 WAT
RNmap scan report for 196.45.52.33
org ) at 2013-05-10 10:24 WAT
RHost is upreport for 196.45.52.33
org ) at 2013-05-10 10:24 WAT
R (0.0011s latency).
MAC Address: 00:17:59:F1:C6:65 (Cisco Systems)
Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds
stems)
```

Fig. 7:interface showing

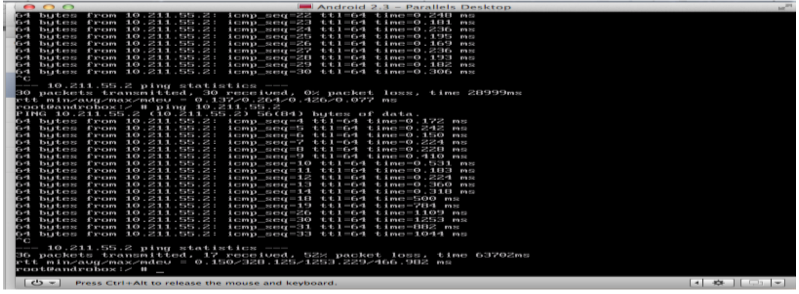
5.1 Result based onPort Scan

When a Ping Sweep was launched on the target host, information about the host were displayed. Several other port scanning techniques was also launched to gather more information from the target host.

FromFig. 7 above, it was observed that the application software program used which executed Linux commands like Nmap at the background was able to gather information that includes AndroidVersion, open, close and filtered ports, status of the AndroidVersion machine if active or down.

5.1 Result based onFlooding

From the test performed in Version 2.3 during flooding attack using the hping3 command as shown in Fig. 8 below. It was observed that before the flooding attack on the target host, ping request from host 10.211.55.2 was responding at 0.122ms and 0.410ms. When the attack was launched, the ping time increased to 500ms and 1044ms and even more as the flooding remained active. This made the target host to drag and responded slowly to the ping request during the flooding attack for the latency was too high.


Android 2.3 – Parallels Desktop
Press Ctrl+Alt to release the mouse and keyboard.

```
64 bytes from 10.211.55.2: icmp_seq=22 ttl=64 time=0.240 ms
64 bytes from 10.211.55.2: icmp_seq=23 ttl=64 time=0.181 ms
64 bytes from 10.211.55.2: icmp_seq=24 ttl=64 time=0.236 ms
64 bytes from 10.211.55.2: icmp_seq=25 ttl=64 time=0.195 ms
64 bytes from 10.211.55.2: icmp_seq=26 ttl=64 time=0.169 ms
64 bytes from 10.211.55.2: icmp_seq=27 ttl=64 time=0.236 ms
64 bytes from 10.211.55.2: icmp_seq=28 ttl=64 time=0.193 ms
64 bytes from 10.211.55.2: icmp_seq=29 ttl=64 time=0.182 ms
64 bytes from 10.211.55.2: icmp_seq=30 ttl=64 time=0.306 ms
^C
--- 10.211.55.2 ping statistics ---
30 packets transmitted, 30 received, 0% packet loss, time 28999ms
rtt min/avg/max/mdev = 0.137/0.264/0.426/0.077 ms
root@androbox:/ # ping 10.211.55.2
PING 10.211.55.2 (10.211.55.2) 56(84) bytes of data.
64 bytes from 10.211.55.2: icmp_seq=4 ttl=64 time=0.172 ms
64 bytes from 10.211.55.2: icmp_seq=5 ttl=64 time=0.242 ms
64 bytes from 10.211.55.2: icmp_seq=6 ttl=64 time=0.150 ms
64 bytes from 10.211.55.2: icmp_seq=7 ttl=64 time=0.224 ms
64 bytes from 10.211.55.2: icmp_seq=8 ttl=64 time=0.228 ms
64 bytes from 10.211.55.2: icmp_seq=9 ttl=64 time=0.410 ms
64 bytes from 10.211.55.2: icmp_seq=10 ttl=64 time=0.531 ms
64 bytes from 10.211.55.2: icmp_seq=11 ttl=64 time=0.183 ms
64 bytes from 10.211.55.2: icmp_seq=12 ttl=64 time=0.224 ms
64 bytes from 10.211.55.2: icmp_seq=13 ttl=64 time=0.360 ms
64 bytes from 10.211.55.2: icmp_seq=14 ttl=64 time=0.318 ms
64 bytes from 10.211.55.2: icmp_seq=18 ttl=64 time=500 ms
64 bytes from 10.211.55.2: icmp_seq=19 ttl=64 time=784 ms
64 bytes from 10.211.55.2: icmp_seq=26 ttl=64 time=1109 ms
64 bytes from 10.211.55.2: icmp_seq=30 ttl=64 time=1253 ms
64 bytes from 10.211.55.2: icmp_seq=31 ttl=64 time=882 ms
64 bytes from 10.211.55.2: icmp_seq=33 ttl=64 time=1044 ms
^C
--- 10.211.55.2 ping statistics ---
36 packets transmitted, 17 received, 52% packet loss, time 63702ms
rtt min/avg/max/mdev = 0.150/328.125/1253.229/466.982 ms
root@androbox:/ # _
```

Fig. 8: hping3 flooding attack on Android 2.3 image during a ping request.

Although, from the analyses and tests carried out, Android operating system faces some challenges. One security flaw in the operating system is that there is no special kind of protection against the handling of flooding attacks, which were quite successful in the simulation as shown in Fig. 8, as the systems were unable to give a response to network during these attacks. From the test performed,Version 2.3 responded to the ping request during the flooding attack but the latency was too high. When the vulnerability test was launched on the open ports, it was observed that due to the privileges granted to the application during installation, the hacker could easily sniff data from the smartphones using any available sniffing tool.

In general opinion of this article, there is an improvement in the security of the Android 4.2 designed for both smartphones and tablet PCs

V. Conclusion

This project work has researched on penetration testing with Android smartphones. It is very interesting to note that Android use on smartphones has increased significantly over the last few years because of the open source platform it adopted with so many features available. These openness and access to the source code has also made hackers to take advantage of these open ports if not properly closed and filtered to attack the smartphones and render them unusable.

In this research work, vulnerability test was performed on three different Versions of Android smartphones. Different tools were used to gather information about a live smartphone system, to know if they are active or inactive, open, closed or filtered.

This paper work reveals that all software has security vulnerabilities. But there are some simple things one can do that will drastically reduce ones exposure and help secure Android smartphones or tablet, as well as protect data. This paper advises that users should protect and secure their smartphones using different ways and not limited to

- Use of password and lock screens to keep smartphones secured.
- Always be in the habit to review apps before installing and check permissions required by the apps Once ports are opened, phones are vulnerable to attack
- Never download apps from unknown sources
- Use an anti-virus or malware app and ensure its is constantly updated.
- Android smartphones is not about apps and trick but about best practices

References

[1] N. Kumar, and M.E UI Haq, *Penetration testing for Android Smartphones*, masters thesis, Chalmers University of Technology, Goteborg, SW, 2011.

[2] F. Alisherov, and F. Sattarova, *Methodology for Penetration Testing*, International Journal of Grid and Distributed Computing, *2(2), 2009*, 44-49.

[3] A. Johnson, K. Dempsey, R. Ross, S. Gupta and D. Bailey, *Guide for Security-Focused configurationManagement of Information System*, (SP800-42), Accessed on May, 2013.

[4] H. Sheikh, J. Cyril, and O. Tomas,*An Analysis of the Robustness and Stability of the Network Stack in Symbian Based Smartphones* Vol. No. 10 2009.

[5] N. Kumar, and M.E UI Haq, *Penetration testing for Android Smartphones*, masters thesis, Chalmers University of Technology, Goteborg, SW, 2011.

[6] Android operating system, (2013). Available from: <http://en.wikipedia.org/wiki/Android_(operating_system) >. [10 March, 2013]

[7] K. Arto, Security *Comparison of Mobile OSes*, and Available from: < http://www.tml.tkk.fi/Opinnot/Tik-110.501/2000/papers/kettula.pdf >. [11 March 2013].

[8] AndroidVersion history, (2013). Available from: <http://en.wikipedia.org/wiki/Android_Version_history>. [15 March 2013]

[9] AndroidVersion, 2013, (2013), Available from: <http://developer.Android.com/guide/basics/what-is-Android.html>. [15 March 2013]

[10] AndroidVersion, 2013, (2013), Available from: <http://developer.Android.com/guide/basics/what-is-Android.html>. [15 March 2013]

[11] Android operating system, (2013). Available from: <http://en.wikipedia.org/wiki/Android_(operating_system) >. [10 March, 2013]

[12] Available from: <http://www.edparsons.com/2007/11/Android-and-lbs-in-the-stack-at-last/> [24 May 2011]

[13] Android-App-Market Available from: <http://www.Android-app-market.com/Android-architecture.html>[30 May, 2013]

[14] F. Alisherov, and F. Sattarova, *Methodology for Penetration Testing*, International Journal of Grid and Distributed Computing, *2(2), 2009*, 44-49