DETECTION OF VULNERABILITY ATTACK THROUGH WIRELESS NETWORK ON ANDROID PLATFORM

MUHAMAD IZHAN FAKRI BIN IMRAN

BACHELOR OF COMPUTER SCIENCE (COMPUTER SYSTEMS & NETWORKING) WITH HONOURS

FACULTY OF COMPUTER SYSTEM AND SOFTWARE ENGINEERING

UNIVERSITI MALAYSIA PAHANG

2014

UNIVERSITI MALAYSIA PAHANG

BORANG PENGESAHAN STATUS TESIS

JUDUL: DETECTION OF VULNERABILITY ATTACK THROUGH WIRELESS NETWORK ON ANDROID PLATFORM

SESI PENGAJIAN: SEMESTER I 2014/2015

SAYA MUHAMAD IZHAN FAKRI BIN IMRAN

Mengaku membenarkan tesis/laporan PSM ini disimpan di Perpustakaan Universiti Malaysia Pahang dengan syarat-syarat kegunaan seperti berikut:

1. Tesis/Laporan adalah hakmilik Universiti Malaysia Pahang.
2. Perpustakaan Universiti Malaysia Pahang dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institut pengajian tinggi.
4. **Sila tandakan (√)

☐	SULIT	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972) *
☐	TERHAD	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan) *
√	TIDAK TERHAD	

Disahkan oleh

[illegible] .... [illegible] ....

Alamat tetap:
LOT 471, LORONG OMAR YUNUS
JALAN BAYAM, 15200 KOTA BHARU
KELANTAN DARUL NAIM

Penyelia:
WAN NURULSAFAWATI BT WAN MANAN

Tarikh: 05/01/2015

Tarikh: 05/01/2015

*Sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali sebab dan tempoh tesis/laporan ini perlu dikelaskan sebagai SULIT atau TERHAD.

STUDENT DECLARATION

I hereby declare that the work in this thesis is my own for quotations and summaries which have been duly acknowledged. The thesis has not been accepted for any degree and is not concurrently submitted for award of another degree.

Signature	: [illegible]
Name	: MUHAMAD IZHAN FAKRI BIN IMRAN
ID Number	: CA11117
Date	: 05/01/2015

SUPERVISOR DECLARATION

I hereby declare that I have checked this thesis and my opinion, this thesis is adequate in terms of scope and quality for the award of the Bachelor's Degree of Computer Science (Computer Systems and Networking) with Honors.

Signature	: [illegible]
Supervisor Name	: MADAM WAN NURULSAFAWATI BINTI WAN MANAN
Date	: 05/01/2015

ACKNOWLEDGEMENT

I am grateful and like to express my sincere gratitude to my supervisor Puan Wan Nurulsafawati Binti Wan Manan for her contribution ideas, invaluable guidance, continuous encouragement and constant support in making this research possible. I appreciate her consistent support from the first day I applied to graduate program to these concluding moments. I am truly grateful for her progressive vision about my training in network engineering, her tolerance of my naive mistakes, and her commitment to my future career. I also sincerely thank for the time spent proofreading and correcting my many mistakes. My sincere thanks to all lecturers and friends at UMP who had helped me in many ways and made my existence at UMP being pleasant and unforgettable. I acknowledge my sincere indebtedness and gratitude to my parents for their love, dream, and sacrifice throughout my life. They always support me mentally and spiritually so that I have the courage to start and complete this thesis until the end.

ABSTRACT

This research is to identify the vulnerability attack that focus on the ARP spoofing attack in the Android platform. This is because each of attacks likes Man-In-The-Middle, Session Hijacking and Password Sniffing are occurring cause of ARP spoofing attack. After that, this research continues with studying about the active technique: ARP request and TCP-SYN injection of packet that intend for detecting the ARP spoofing attack. This technique is better than the passive technique that also used for detection of the ARP spoofing. The passive technique is not effective to use because it takes time too long to detect and report the attack. Additionally, if the MAC addresses flooding into the ARP cache on switch with ARP requests and replies may lead to new report tables every time which makes it is unreliable. Next, at the end of this research will develop an Android application for applying the active technique as a method to detect and prevent the ARP spoofing attack on Android smartphone. The Android application will be developed by using Java programming language. The main aim here is that Android application can avoid the attacker steal the Android user's information by launching some attacks that started with the ARP spoofing attack.

ABSTRAK

Kajian ini adalah untuk mengenalpasti kelemahan serangan yang mengfokuskan pada serangan penipuan ARP dalam Android platform. Ini adalah kerana setiap daripada serangan seperti Man-In-The-Middle, Session Hijacking dan Password Sniffing adalah berlaku disebabkan serangan penipuan ARP. Selepas itu, kajian ini diteruskan dengan mempelajari tentang teknik aktif: ARP Request dan suntikan TCP-SYN paket yang bertujuan untuk mengesan serangan penipuan ARP. Teknik ini adalah lebih baik daripada teknik pasif yang juga digunakan untuk mengesan penipuan ARP tersebut. Teknik pasif tidak berkesan untuk digunakan kerana ia mengambil masa terlalu lama untuk mengesan dan melaporkan serangan itu. Di samping itu, jika alamat MAC membanjiri ke dalam ARP Cache pada suis rangkaian dengan ARP Request dan ARP Reply yang boleh membawa kepada laporan jadual baru setiap masa yang menjadikan ia tidak boleh dipercayai. Seterusnya, pada akhir kajian ini akan dibangunkan sebuah aplikasi Android untuk mengaplikasikan teknik aktif sebagai kaedah untuk mengesan dan mencegah serangan penipuan ARP pada telefon pintar Android. Aplikasi Android itu akan dibangunkan dengan menggunakan bahasa pengaturcaraan Java. Tujuan utama di sini adalah bahawa aplikasi Android itu boleh mencegah penyerang yang mencuri maklumat pengguna Android dengan melancarkan beberapa serangan yang dimulai dengan serangan penipuan ARP.

TABLE OF CONTENTS

	PAGE
STUDENT DECLARATION	II
SUPERVISOR DECLARATION	III
ACKNOWLEDGEMENT	IV
ABSTRACT	V
CONTENTS	VII
LIST OF FIGURES	X
LIST OF TABLES	XI

SECTION	**CONTENTS**	**PAGE**
CHAPTER 1		1
INTRODUCTION		1
1.1	INTRODUCTION	1
1.2	PROBLEM STATEMENT	3
1.3	OBJECTIVE	3
1.4	SCOPES	4
CHAPTER 2		5
LITERATURE REVIEW		5
2.1	INTRODUCTION	5
2.2	THE EXISTING TECHNIQUES FOR DETECTION AGAINST ARP SPOOFING	7
2.2.1	Passive Techniques for Detection against ARP Spoofing	7
2.2.2	Active Techniques for Detection against ARP Spoofing	8
2.3	COMPARISONS BETWEEN PASSIVE AND ACTIVE TECHNIQUES ON DETECTION OF ARP SPOOFING	9
2.4	PROTECTION METHODS AGAINST ARP SPOOFING	10
2.4.1	ARP Cache Updating Method	10
2.4.2	Configure the Static ARP Cache	10
2.4.3	Controlling With Switching Equipment	10
2.4.4	Information Encryption	11
2.4.5	Checking the ARP Cache Periodically	11

2.5 CONCLUSION 11

CHAPTER 3 12

METHODOLOGY 12

3.1 INTRODUCTION 12

3.2 RESEARCH METHODOLOGY 13

3.2.1 Planning Phase 14

3.2.2 Analysis Phase 14

3.2.3 System Design Phase 15

3.2.4 Implementation Phase 21

3.2.5 Testing Phase 21

3.3 ANALYZING SOFTWARE AND HARDWARE REQUIREMENTS 22

3.3.1 Software Requirements 22

3.3.2 Hardware Requirements 23

3.4 DESCRIBE HARDWARE AND SOFTWARE USES IN THE RESEARCH 24

3.4.1 Eclipse IDE with Built-In ADT 24

3.4.2 Android Software Development Kit (SDK) 24

3.4.3 dSploit 24

3.4.4 Arpspoof 25

3.4.5 Ninetology Black Pearl II 25

3.4.6 Samsung Galaxy Ace Plus 25

3.4.7 Acer Aspire 4755G 25

3.5 GANTT CHART 26

3.6 LIMITATION OF THE RESEARCH 26

3.7 SUGGESTIONS FOR IMPROVEMENT OF FUTURE RESEARCH 26

3.8 CONCLUSION 27

CHAPTER 4 28

DESIGN AND IMPLEMENTATION 28

4.1 SYSTEM ENVIRONMENT ARCHITECTURE 28

4.2 MAIN INTERFACE 30

4.2.1 Main Interface's Checkboxes 31

4.2.2 Main Interface's Buttons 34

4.2.3 Main Interface's Seek Bar 35

4.3 ALERT INTERFACE 37

4.3.1 Alert Interface's Text View 38
4.3.2 Alert Interface's Buttons 39
4.4 DETECTION ALGORITHM AGAINST ARP SPOOFING 41
4.5 CONDITIONS AND FORMULA FOR DETECTION OF ARP SPOOFING 44
4.6 PROTECTION TECHNIQUE AGAINST ARP SPOOFING 45
CHAPTER 5 46
RESULTS AND DISCUSSION 46
5.1 INTRODUCTION 46
5.2 RESULT ANALYSIS 46
5.3 CONSTRAINTS 48
5.3.1 Development Constraints 48
5.3.2 System Constraints 49
5.4 ENHANCEMENT FOR FUTURE RESEARCH 49
CHAPTER 6 50
CONCLUSION 50
6.1 CONCLUSION 50
REFERENCES 51
APPENDIX A 52
GANTT CHART 52

LIST OF FIGURES

FIGURE **TITLE** **PAGE**

Figure 1 The Existing Tools for Defending against ARP Spoofing. (Wikipedia, 2012)........6
Figure 2 Waterfall Model 13
Figure 3 Use Case Diagram........16
Figure 4 First Interface 17
Figure 5 Second Interface........18
Figure 6 System Flowchart........20
Figure 7 System Environment Architecture 29
Figure 8 Main Interface 30
Figure 9 Main Interface's Checkboxes 32
Figure 10 Main Interface's Buttons........34
Figure 11 Main Interface's Seek Bar 36
Figure 12 Alert Interface........37
Figure 13 Alert Interface's Text View 38
Figure 14 Alert Interface's Buttons........39
Figure 15 The Algorithm of Detection Process........45
Figure 16 The Gantt Chart of Workflow 52

LIST OF TABLES

TABLE TITLE PAGE

Table 1 Components of First Interface..19
Table 2 Components of Second Interface..21
Table 3 Software Requirements..22
Table 4 Hardware Requirements...23
Table 5 Types of Conditions in ARP Cache..44

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Latterly, various smartphone models were expelled in the market, which uses various types of operating system (OS) like iOS, Windows Mobile, Blackberry and Android. Each of the OS has their application distinctively and can be classed to types of application, paid-up or not, and first-party or third-party. The smartphone OS that has become the most popular operating system today is the Android operating system. According to (Barra, H, 2012), "There are 500 million devices active globally and over 1.3 million added every single day."

As fast as the technology has grown today, everything can be accessed on our smartphone from the biggest matter until the smallest matter by pressing only one button to show up the information. But certain things not all data should be shared for everyone but today the problem is coming out when someone trying to get data without permission from its owner. And worse than that, he/she alter the data or information and send to somebody to get benefit from that data. Normally, the usual technique to attack someone on the same network is by using ARP spoofing or ARP poisoning. "ARP spoofing is a technique that allows attacker to send spoofed Address Resolution Protocol (ARP) messages onto a Local Area Network (LAN)" (Wikipedia, 2012). ARP is a network protocol that uses to communicate between the end devices in LAN. ARP will mapping the IP address and MAC address of devices together and keep it in the ARP cache as reference to do communication. The attacker will poison the ARP cache which makes changes to the IP-MAC mapping and proceeding with other attacks. For example, Man-In-The-Middle (MITM), Session Hijacking, Password Sniffing and etc.

The detection application against ARP spoofing is important to use in order the attacks can be detected and immediately prevent it from a user's smartphone. On the other hand, if let it be on a user's Android smartphone, many of their essential information being used or modified without their permission. This situation will be worse when the company suffers losses due to secret information about companies leak invaded.

So, the contribution of the research will develop an Android application which can provide detection and protection against ARP spoofing by installing the application on Android smartphone due to the weak security on a private wireless network. If the smartphone getting spoofed, the Android application will notify to the user quickly in the notification bar. The user can prevent the attack by choosing to terminate the Wi-Fi connection immediately in order to protect the privacy information gets stolen. Moreover, the Android application is user friendly whereas it have user interface and user do not need to type commands to run it.

In conclusion, the application that provides security for Android smartphone against ARP spoofing on a private wireless network. So that, it can decrease the number of victims from hijacking or spoofing attacks on private wireless networks. Besides, the user can use their smartphone without worried because everything in safe and sound being protected by this application.

1.2 PROBLEM STATEMENT

Almost of people do not have enough knowledge about the ARP spoofing, which how to detect it and protect their smartphone from affected of it. This matter is dangerous and make their information can be stolen easily by someone that have bad intentions. In a private wireless network, the attacker can exploit into it and do the ARP spoofing attack by applying the ARP cache poison into the victim's wireless access point as a step to launch other attacks. For example, man-in-the-middle attack that use to steal victim's private information for something illegal uses.

Besides, the existing application that was developed today need to root in order it functioning well. Additionally, the number of security application is still less in the market and almost of them don't have the user interface and it becomes really unfriendly to users. Furthermore, in Google Play Store the users need to pay for downloading the security application and it's more expensive for the user.

1.3 OBJECTIVE

i. To study the vulnerability of ARP spoofing attack on a private wireless network.

ii. To apply the active technique on detection of ARP spoofing attack for Android devices.

iii. To develop ARP spoofing detection prototype for Android users who connect their Android devices on a private wireless network.

1.4 SCOPES

i. The research is focusing on detection of ARP spoofing attack.
ii. The application uses for detecting ARP spoofing in a private wireless network.
iii. Targeted user is for Android users only.

CHAPTER 2

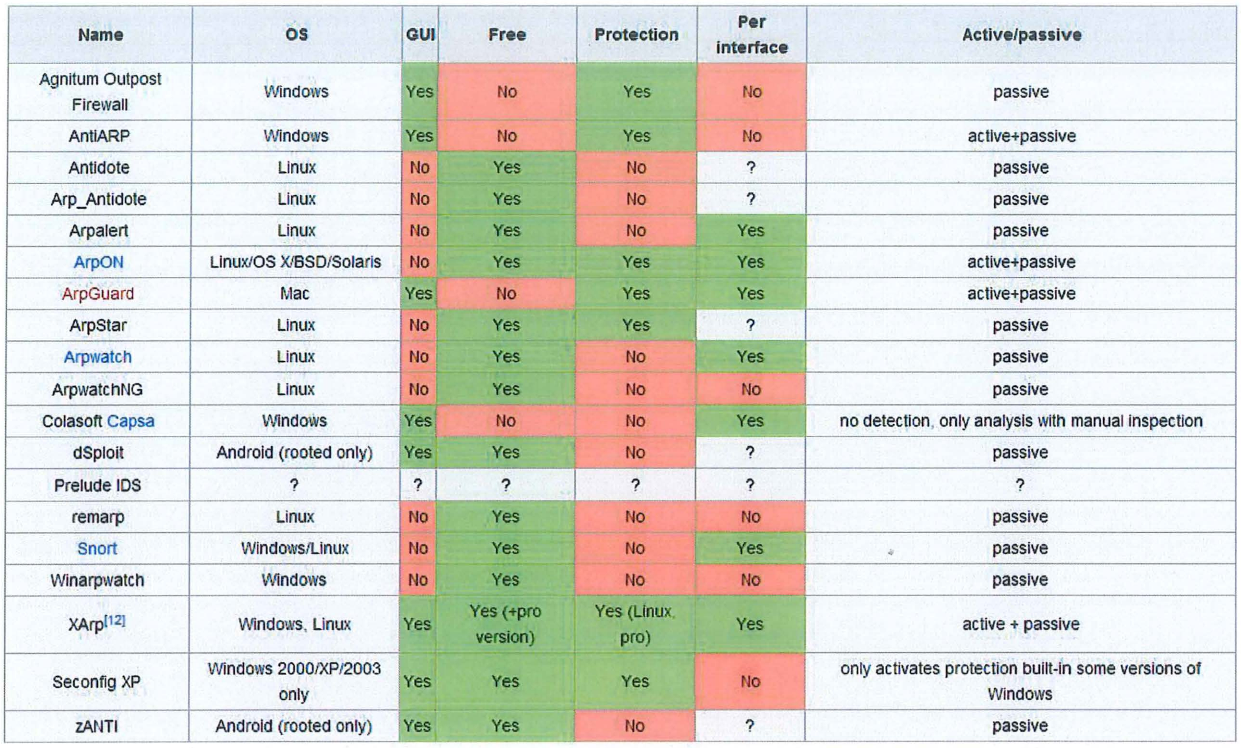
LITERATURE REVIEW

2.1 INTRODUCTION

"ARP stands for Address Resolution Protocol. ARP acts as a network layer over the Internet Protocol address (IP) and converts it into a Media Access Control address (MAC address) or Ethernet Hardware Address (EHA)." (Kumar, V, 2012). ARP spoofing is a type of attack which attacker can change the MAC address. It works on wired or wireless local network. Many hackers like ARP spoofing attack because it is very simple to apply.

This research is focusing on detection of the vulnerability of ARP spoofing attack. Most public people do not know what is the vulnerability? For the information, vulnerability means any weakness or flaw existing in a system or the susceptibility of a system to a specific threat attack or harmful event or the opportunity available to a threat agent to mount that attack.

The security problems of a system are the result of vulnerabilities. Vulnerabilities are the weaknesses of the system that can be exploited in ways that violate network policies (Rebecca Curley Bace, 2005). The vulnerabilities can take on different forms, for example, a vulnerability may occur in the design and implementation of software and hardware components of a system. There are a lot of defense against ARP spoofing and different techniques or tools work on different operating system. For example:



Name	OS	GUI	Free	Protection	Per interface	Active/passive
Agnitum Outpost Firewall	Windows	Yes	No	Yes	No	passive
AntiARP	Windows	Yes	No	Yes	No	active+passive
Antidote	Linux	No	Yes	No	?	passive
Arp_Antidote	Linux	No	Yes	No	?	passive
Arpalert	Linux	No	Yes	No	Yes	passive
ArpON	Linux/OS X/BSD/Solaris	No	Yes	Yes	Yes	active+passive
ArpGuard	Mac	Yes	No	Yes	Yes	active+passive
ArpStar	Linux	No	Yes	Yes	?	passive
Arpwatch	Linux	No	Yes	No	Yes	passive
ArpwatchNG	Linux	No	Yes	No	No	passive
Colasoft Capsa	Windows	Yes	No	No	Yes	no detection, only analysis with manual inspection
dSploit	Android (rooted only)	Yes	Yes	No	?	passive
Prelude IDS	?	?	?	?	?	?
remarp	Linux	No	Yes	No	No	passive
Snort	Windows/Linux	No	Yes	No	Yes	passive
Winarpwatch	Windows	No	Yes	No	No	passive
XArp[12]	Windows, Linux	Yes	Yes (+pro version)	Yes (Linux, pro)	Yes	active + passive
Seconfig XP	Windows 2000/XP/2003 only	Yes	Yes	Yes	No	only activates protection built-in some versions of Windows
zANTI	Android (rooted only)	Yes	Yes	No	?	passive

Figure 1 The Existing Tools for Defending against ARP Spoofing. (Wikipedia, 2012)

In Android devices, there are some applications for ARP spoofing tools that using GUI to launch an attack on the network. The attacker does not need skills or knowledges about the ARP spoofing to launch attack. Therefore, there are some techniques used to detect and protect against ARP spoofing.

2.2 THE EXISTING TECHNIQUES FOR DETECTION AGAINST ARP SPOOFING

Below are the two existing techniques used in the detection of ARP spoofing attack.

i. Passive Technique
ii. Active Technique

2.2.1 Passive Techniques for Detection against ARP Spoofing

The passive techniques to improve the detection of session hijacking attack are cheap, dependable, and have less effect on network performance. There are:-

I. **Monitoring Received Signal Strength**. "By monitoring the values of RSS for a particular STA or an AP from a passive monitor can make a dynamic profile for the communicating nodes based on their RSS values. Any abnormal changes can be dropped as suspicious action indicative of a potential session hijacking attack." (Gill, Rupinder S. and Smith, Jason and Looi, Mark H. and Clark, Andrew J., 2005).

II. **Monitoring Round Trip Times of RTS-CTS Hand shake**. "The Timerrt between two nodes change fast and disconnect can use to observe session hijacking attacks. Interestingly, this property still remains usable if rather than monitoring Timertt s values on the sender; a passive static wireless monitor is used for these time measurements. However the monitor cannot compute Timertt s completely as it is a property relative to the sender." (Gill, Rupinder S. and Smith, Jason and Looi, Mark H. and Clark, Andrew J., 2005).

2.2.2 Active Techniques for Detection against ARP Spoofing

ARP request and TCP SYN packets

I. **Full ARP Cycle**: "If spoofing is active, the original ARP request will reply by attacker and the real host and for the same IP, we can notice a conflict in the MAC address. If the source address of both the ARP replies received the TCP SYN packet. The attacker's customized stack replies as well along with the real host; two TCP packets in response will send to us. By doing so, can easily to detect spoofing." (Ramachandran, V., and Nandi, S, 2005).

II. **Request Half Cycle**: "By sending one ARP request packet back to the sender and check the reply(s) for spoofing, we will try to authenticate the sender of the request. We will raise a Spoof Alarm if the MAC address in the ARP Request Half Cycle packet does not match to the source MAC addresses in the ARP reply packet received for the injected ARP." (Ramachandran, V., and Nandi, S, 2005).

III. **Reply Half Cycle**: "Get multiple replies to the ARP request send if customized stack is used. Also will get multiple TCP (SYN/ACK or RST) packets in return with different MAC address after send the TCP SYN packets out to the sources of the ARP request. This is sufficient to reason out that a spoofing is activating." (Ramachandran, V., and Nandi, S, 2005).

2.3 COMPARISONS BETWEEN PASSIVE AND ACTIVE TECHNIQUES ON DETECTION OF ARP SPOOFING

(Ramachandran, V., and Nandi, S, 2005) proved that "their technique is much faster and reliable than the passive detection techniques. This technique can be used in a large network and it will immediately detect ARP spoofing attacks, even if the attack had begun before the tool using this technique was operational. This is because the time lag between learning and detection is very less as we probe the authenticity of hosts as soon as we see ARP traffic from them. This technique also verifies the authenticity of the ARP traffic on the network and does not blindly add newly seen traffic to its database. Even in the event of an actual attack this technique can detect the correct IP to MAC address mapping of the real host in the absence of the attacker using a customized network stack. If the attacker uses a customized stack, which replies to these probes we are still able to detect ARP spoofing, but will not be able to predict the real MAC to IP address mapping. So even in the worst-case scenario (in the presence of a customized stack) the performance is still better than using a Passive detection technique."

2.4 PROTECTION METHODS AGAINST ARP SPOOFING

2.4.1 ARP Cache Updating Method

(Liu, Y., Dong, K., Dong, L., and Li, B., 2008) stated that "For the ARP protocol receiving the ARP response and updating the cache without ARP request, this has created the ARP spoofing condition, so we can formulate the ARP cache updating rules. It stipulates that the sequence of receiving ARP protocol is fist, sending out ARP request, then receiving the matching ARP response package, these non-response or non-matching response package will be deleted, this can prevent the ARP spoofing efficiently."

2.4.2 Configure the Static ARP Cache

There are a few methods to guard against ARP spoofing. According to (Liu, Y., Dong, K., Dong, L., and Li, B., 2008) "One important precondition of ARP spoofing is that the ARP cache is dynamically changed, if the network manager makes a registration of IP address and MAC address and prevent the dynamically ARP cache updating. When the spoofing ARP response to the host it cannot update the ARP cache, so it cannot achieve the goal of spoofing. But the expense is bigger and easy mistaking."

2.4.3 Controlling With Switching Equipment

(Liu, Y., Dong, K., Dong, L., and Li, B., 2008) stated that "Using the switchboard can make the physical of network into subsection, and bind the IP address with the MAC address statically, the switchboard will compare the source address to the port report of the source address information when a port received the message, if changes occurred, then forbids automatically to the port until the conflict solved. Making use of the insulation function of the router, make the ARP beguilement."

2.4.4 Information Encryption

(Liu, Y., Dong, K., Dong, L., and Li, B., 2008) stated that "Based on the ARP spoofing theory, Sniffing is difficult to be checked. If both of the communication side encrypted the message, even though the message was caught by the attacker it cannot get the useful message without the decryption algorithm, so it guarantees the network transmitting security."

2.4.5 Checking the ARP Cache Periodically

According to (Liu, Y., Dong, K., Dong, L., and Li, B., 2008) "The network manager catch the ARP request and the ARP response periodically, check the reliability of the ARP response, Taking turns periodically, check the reflection between the host's ARP cache IP address and the MAC address."

2.5 CONCLUSION

In conclusion, the technique can be used in the research is comparing the IP addresses and MAC addresses as a method to detect against ARP spoofing attack although there is a lot of efficient techniques to apply. It makes comparison between the devices' IP address and MAC address which send an ARP request to the users' device with the IP and MAC address of router or gateway. For protection against ARP spoofing attack, the termination of the network connection of the device is chosen when it detects the ARP spoofing on the network. This is because it is the safest way to avoid Android users from getting the ARP spoofing attack. Although this is not a fully protected compared to others because the application is built for Android platform so there are some limitations for using another technique to build it if compared to Windows OS, which is the biggest OS in the world and almost all techniques are compatible to it.

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

The system development methodology that used in the research is based on the waterfall model. This methodology is a series of steps to develop the research from beginning until the end. All the phases in waterfall model that appear in the research need to explain briefly. The methodology may include present and previous information needed to collect based on the research requirements. Besides that, the user interface of Android application that was designed will be elaborate in this chapter. The system flowchart of the application is drawn and explained in detail. In this case also the selection of software and hardware to be used should be decided carefully, which must be compatible and available for development stage. Finally, at the end of this chapter the Gantt chart for the overall research process is framed which list out all the activities have been done and will be done.

3.2 RESEARCH METHODOLOGY

The waterfall model is a conceptual model that used in the research that can describe the steps or phases involved in the information system development of the research. This waterfall model consists of planning, analysis, system design, implementation and testing of the completed application. Besides that, the waterfall model is also a systematic model for the problem solving, this is because that it can be seen the output of each stage becomes the input for the next step or phase.

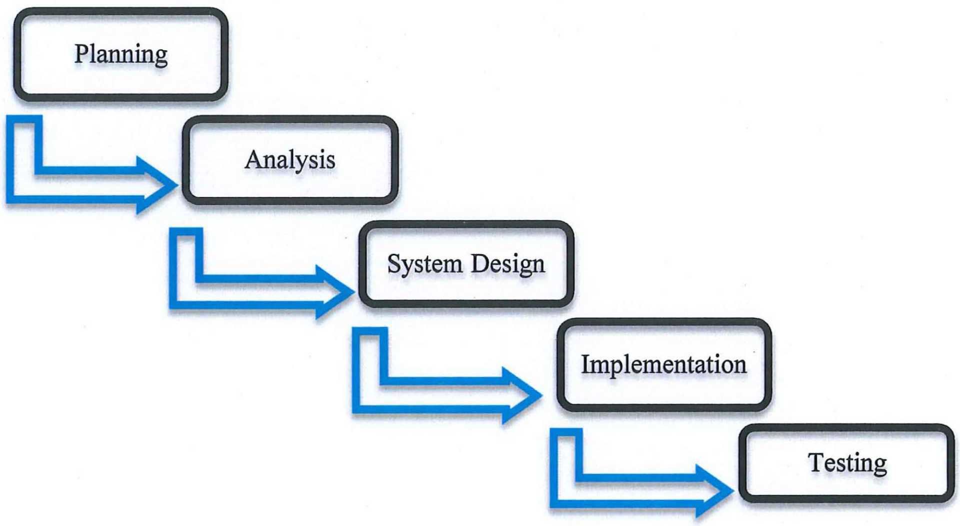


Figure 2 Waterfall Model

3.2.1 Planning Phase

The first phase of waterfall model is planning phase. In the phase is the most critical step in completing the overall research. Brainstorming is the most important in the initial stages for this system. Contribution of idea is done in this phase; the selection of the research title for this study is committed before continuing with other tasks. Nowadays, almost everyone has their smartphone and the Android operating system is chosen as a platform for the smartphone. Because so many choose this type of smartphone, the application developer for it is increase and related crimes for it also increased. That is why the detection of ARP spoofing on Android devices is selected as the title. After title selected, need to assemble the requirements of the application involved with the lowest cost due to the restriction of budget such as hardware, software and tool. With the careful planning in this step, is necessary to coordinate activities and reduce the management of project risks effectively. In this phase, the problem statements, objectives and the scope of this research were determined.

3.2.2 Analysis Phase

In the analysis phase, all information that was collected from the previous phase need to analyst whether compatible with research requirements and valid for development of the application. If some information is not compatible for the research, so it needs to find another to replace them. All collected information will be finalized and choose any one that suitable to develop the application. Examples of the requirements to be examined are as follows, which are Acer Aspire 4755G laptop, Ninetology Black Pearl II, Samsung Galaxy Ace Plus, Eclipse IDE and etc. All the requirements are chosen because it fits to each other and able to achieve the objectives in this research. Besides, the research about the chosen technique for the application is analyzed which are for monitoring ARP Cache of devices and comparing the mapping of IP-MAC addresses.

3.2.3 System Design Phase

In this phase, all the analysis data will be converted into the form of data, such as flow charts, diagrams, and design. All this is to facilitate understanding of this research, it differs from others in order not to trip off of the platform, and shows an overview of how it works in more detail and develops the application according to the requirements stipulated. Firstly, the use-case diagram is designed to determine the interaction between a role and an application to achieve the goal. Additionally, the user interface also designed using Eclipse IDE software with built-in ADT. The function that was created is also explained for each user interface. After that, the flowchart of the application is explained in that designed step by step.

3.2.3.1 Use Case Diagram

Android user can use the application function to detect and protect against ARP spoofing on Android devices, whereas developer of that application can maintenance and update the application. Android User cannot maintenance the application by himself or herself. Besides that, Android user also cannot update the application in the Google Play Store by him or her because due to the source code and layout coding is in the developer.

The use-case diagram is used to explain how the detection of ARP spoofing attack is flow. In the use-case diagram below, the user play a role sends request top server responds by providing the requested service. Besides that, the administrator is alerted by the detection of the vulnerability attack Android application of any suspicious internet protocol address or whenever the intrusion is detected.

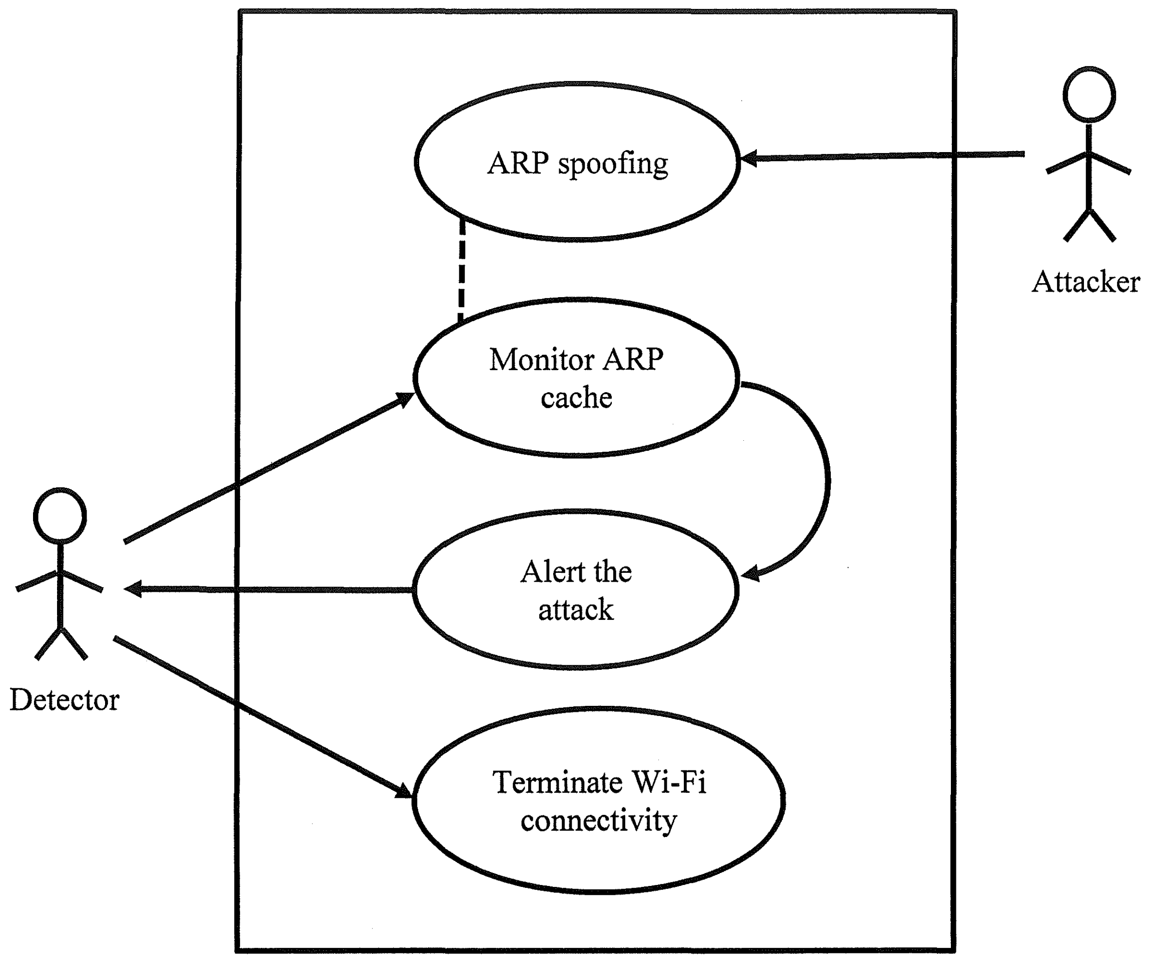
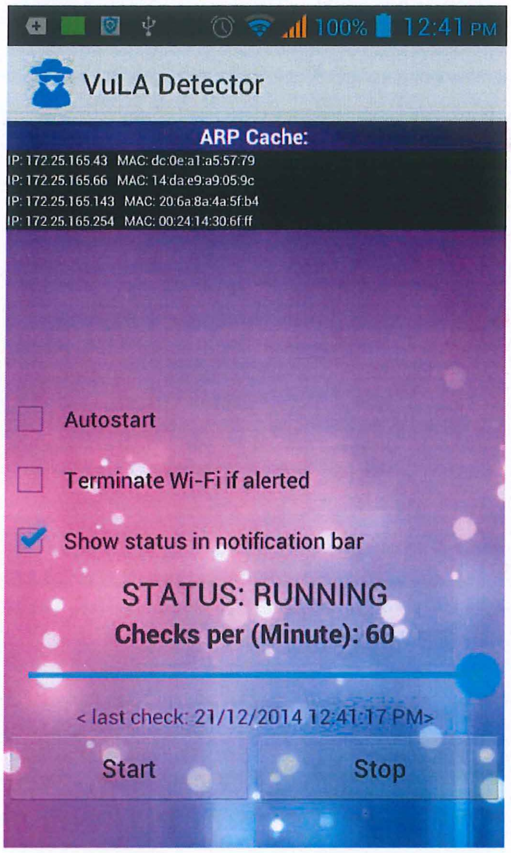


Figure 3 Use Case Diagram

3.2.3.2 User Interface

The user interface of the application is very user friendly. This application is very simple, which it does not need to login or sign up when to use it. What the task that needs to do in order its functioning connects to the Wi-Fi connection and then start the application by pressing the start button for starting the detection process. The stop button is used for stopping the detection process against ARP spoofing attack. When an ARP spoofing attack is detected, then the application will pop up into the new interface that show the information about the attacker and give the user some option to prevent from it.



ARP Cache:

```
IP: 172.25.165.43   MAC: dc:0e:a1:a5:57:79
IP: 172.25.165.66   MAC: 14:da:e9:a9:05:9c
IP: 172.25.165.143   MAC: 20:6a:8a:4a:5f:b4
IP: 172.25.165.254   MAC: 00:24:14:30:6f:ff
```

☐ Autostart

☐ Terminate Wi-Fi if alerted

☑ Show status in notification bar

STATUS: RUNNING

Checks per (Minute): 60

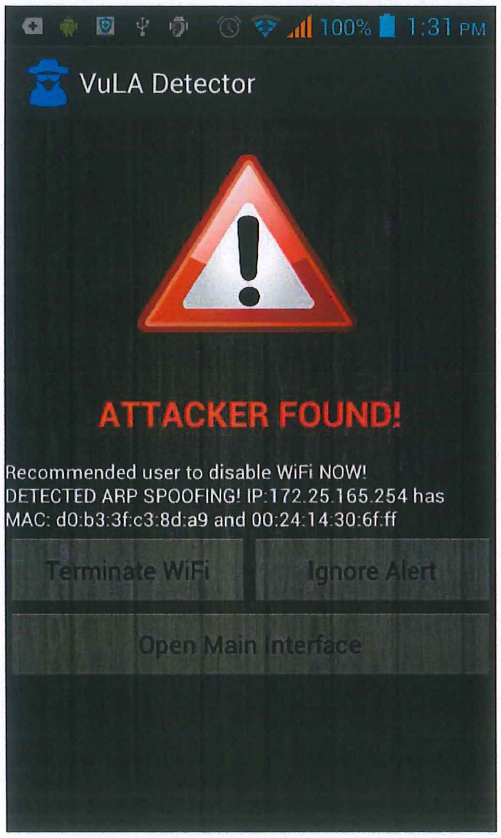
< last check: 21/12/2014 12:41:17 PM>

Figure 4 First Interface

Table 1 Components of First Interface

Components of Interface	**Description**
List View (ARP cache)	To show a list of the mapping IP-MAC addresses in the ARP Cache.
Checkbox (Autostart)	When the Wi-Fi connection is on, the application will start automatically.
Checkbox (Terminate Wi-Fi if alert)	If the attack is detected, the application will terminate the Wi-Fi connection automatically.
Checkbox (Show status in notification bar)	The status of the application will show in

	the notification bar on top of the smartphone's screen.
Text View (Status)	To show what the status is going on the application either the detection process is running or not.
Seek Bar	To set the time interval for checking the ARP Cache between 1-60 minute.
Button (Start)	Activate the detection process.
Button (Stop)	Deactivate the detection process.



ATTACKER FOUND!

Recommended user to disable WiFi NOW!
DETECTED ARP SPOOFING! IP:172.25.165.254 has MAC: d0:b3:3f:c3:8d:a9 and 00:24:14:30:6f:ff

Figure 5 Second Interface

Table 2 Components of Second Interface

Components of Interface	**Description**
Text View (Recommended user…)	To show the information about the ARP spoofing attack (IP address and MAC address)
Button (Terminate Wi-Fi)	To terminate Wi-Fi connection.
Button (Ignore Alert)	Just ignore the attack.
Button (Open Main Interface)	Back to main interface.

3.2.3.3 Flowchart

The general process of the application shows the flowchart as follows below. The user can start monitoring the ARP cache by clicking start button on the Main Interface. The monitoring process will run for ensuring IP-MAC mapping either correct or not. If one IP address has two different MAC address, thus it suspected occurrence of ARP spoofing in the network. So, the application will give alert to users by showing out the Alert Interface that contains information about the attack. After that, the user was given two options, either to terminate the Wi-Fi connectivity for preventing ARP spoofing or the user just ignores the alert.

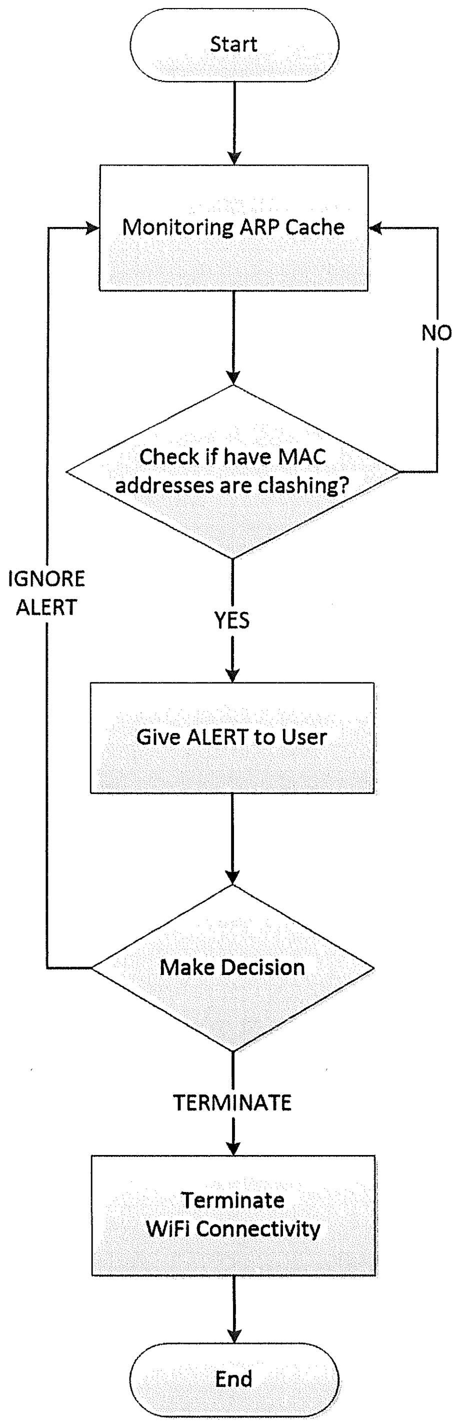


Figure 6 System Flowchart

3.2.4 Implementation Phase

After system design phase is done, the research continues with coding the application by using Java language in Eclipse IDE software. Some applications' source codes are taken as reference to improve the application. Basically, the way to code is learn from some website like Youtube.com and Android.com which have a lot of video and tutorials that can be guideline to develop the application. The application is developed by using API level 8 until 17 which mean this application is compatible for Android 2.3 (Froyo) until Android 4.2 (Jellybean).

3.2.5 Testing Phase

After the coding of the application is completed, the coding will be compiled into an APK file and install it into a smartphone. The testing is proceeding whereas the application will be tested to the real scenario. During the testing process, any bugs and errors will be tested out to ensure all problems can be solved before releasing for public use. The application must be successfully run on Android 2.2 until Android 4.2 and all functions should be tested as well. After that, the performance test, responsiveness, availability, robustness, reliability, scalability, resource usage, durability and stability of the system are tested under a particular workload. For system testing, the whole system is tested that will be started from the smallest function to the biggest function to ensure all modules work together and it perform well as an application.

3.3 ANALYZING SOFTWARE AND HARDWARE REQUIREMENTS

The analysis of this matter includes like a software, hardware and programming language that were used in those previous systems and also what is the suitable used in developing the detection of the ARP spoofing attack system through a wireless network on Android-based devices. Therefore, there is some software and hardware is chosen.

3.3.1 Software Requirements

Table 3 Software Requirements

Software	**Description**
Eclipse IDE with built-in ADT	Eclipse + ADT plugin Android SDK tools Android Platform-tools The latest Android Platform The latest Android System Image for the emulator
Android Software Development Kit	Create the Android application
dSploit.apk	ARP spoofing & Session Hijacking attack Password sniffer
Arpspoof.apk	ARP spoofing

The Android interface is being designed and managed using Eclipse. Eclipse is an editor for Java Script programming language for Android, but it also can use to design the interesting user interface for Android application. Using this Eclipse software for the development of the detection of the vulnerability attack Android application it becomes faster and easier. The dSploit and Arpspoof Android application use to attack the detection Android application as for the testing process.

3.3.2 Hardware Requirements

There are few hardware are used in the Android application for a private wireless network. It is included wireless access point, laptop, smartphones and UTP cables as communication media.

Table 4 Hardware Requirements

Hardware	**Description**
Ninetology Black Pearl II	Android 4.0 (Jelly Bean) 480 x 800 pixels, WVGA Dual Core 1.0 GHz 512MB RAM
Samsung Galaxy Ace Plus	Android 2.3 (Gingerbread) 320 x 480 pixels, WVGA 1 GHz Cortex-A5 512 MB RAM
Acer Aspire 4755G (Laptop)	Window 8.1 Professional 64-bits Intel® Core i5 (2nd Gen) 2410M @ 2.3 GHz 8GB DDR3 RAM NVIDIA® GeForce® GT 540M 2GB
TP-LINK TL-WA901ND (AP)	Wireless Standards : IEEE 802.11n, IEEE 802.11g, IEEE 802.11b Wireless Security : 64/128/152-bit WEP/WPA/WPA2,WPA-PSK/ WPA2-PSK

3.4 DESCRIBE HARDWARE AND SOFTWARE USES IN THE RESEARCH

All the software and hardware will be described briefly and give some reasons why it be chosen to use in the research.

3.4.1 Eclipse IDE with Built-In ADT

Eclipse IDE is chosen because it is software to develop the Android application that's really easy to use. This Android developer software is written by the Java language. The users just drag and drop what item or widget that he want to use and insert some codes to the Android application. Besides, it is open source software that users do not to pay to use it.

3.4.2 Android Software Development Kit (SDK)

It is used to run the Android application in a virtual Android platform. This software is needed because to test the Android application whiles it in the developing process.

3.4.3 dSploit

It is an Android application that will be installed in Ninetology Black Pearl II and uses to do ARP spoofing attacks on the same network or private wireless network. This application is free that can be downloaded in Google Play Store. Besides, it needs a rooted smartphone in order to run it. Some attacks in dSploit such as MITM, session hijacking, password sniffing and etc.

3.4.4 Arpspoof

This application is used to launch the ARP spoofing attack to someone. This application also is a free application that can be downloaded in Google Play Store. Arpspoof and dSploit are used to test the Android application developed to find some defects or errors in there.

3.4.5 Ninetology Black Pearl II

Ninetology Black Pearl II with Android 4.0.1 Jellybean is used as an attacker. ARP spoofing and session hijacking applications will be run into it to perform attacks against Samsung Galaxy Ace Plus which acts as a defender on a private wireless network.

3.4.6 Samsung Galaxy Ace Plus

Samsung Galaxy Ace Plus with Android 2.3 Gingerbread is used as a defender. The developed application will be installed into it and test run to find the bugs in the application. Then, this smartphone will protect against ARP spoofing attacks from other Android devices using the application. If the application runs successfully, then the application can release to the market with a minimum bug within it.

3.4.7 Acer Aspire 4755G

Acer Aspire 4755 with Windows 8.1 is being used to develop Android applications due to the capability of Windows 8.1. Besides that, this laptop is more portable than desktop therefore it can bring from place to place to do my research.

3.8 CONCLUSION

There are all three chapters together in the Final Year Project 1 thesis. The chapters are Introduction, Literature Review and Methodology. Each chapter explains the system development progress and the development process of the detection of vulnerability attack system.

This project development main objective is to develop a prototype for a detection system from ARP spoofing attack and it will be implemented on the Android platform through the private wireless network, especially home-network that using the wireless access point. Development of this system is based on the active technique the system use and the Java language to design the interface.

Keep in mind, the application is applicable in real life. The users are able to download and install the Android application into their device, but it's only limited for smartphones that running Android 2.2 Froyo until Android 4.2 Jellybean. Hence, the application is limited to Android user only.

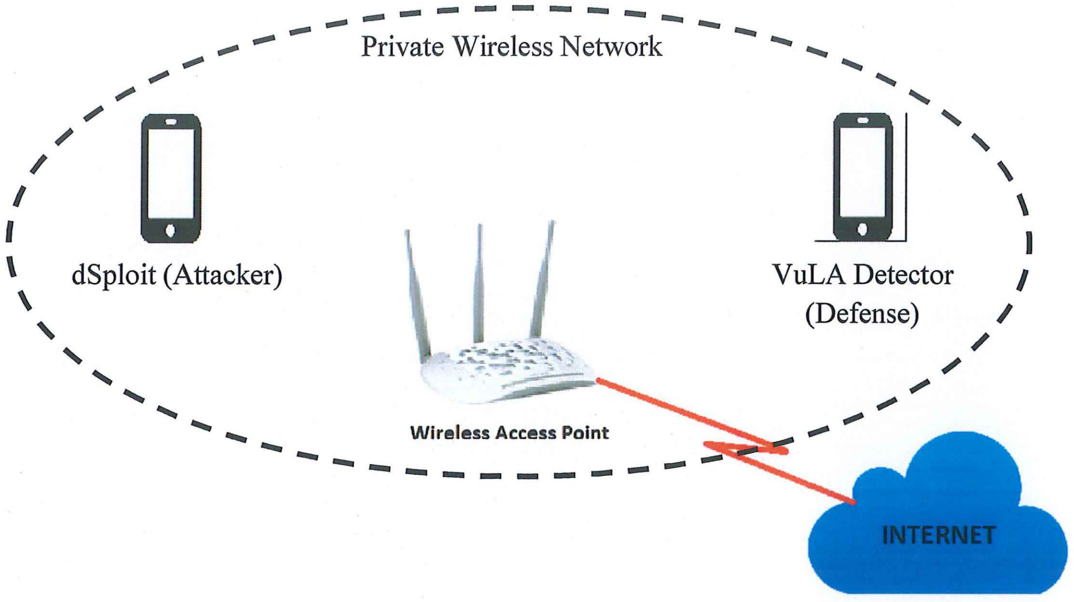
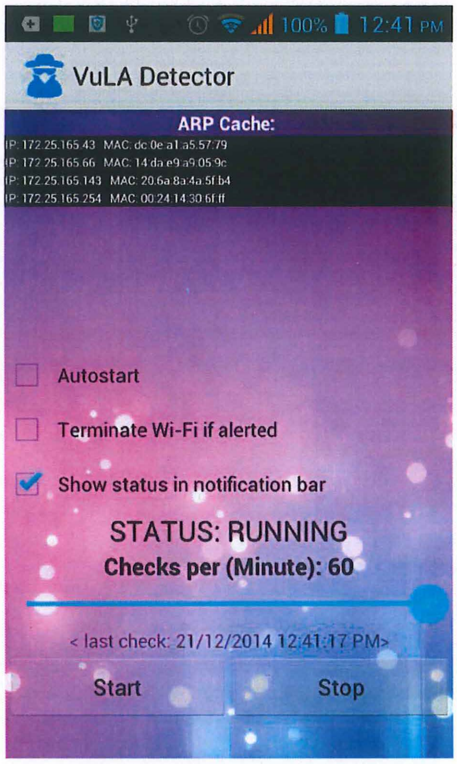


Figure 7 System Environment Architecture

4.2 MAIN INTERFACE

Once the user or targeted user install and run the VuLA Detector Android application on the Android platform. The network connection should be enabled before launching the VuLA Detector application. The users need to turn on the Wi-Fi Android device and connect to user's own wireless access point. This is the main interface for VuLA Detector which have 1 List View, 3 Check Boxes, 4 Text View, 1 Seek Bar and 2 Buttons.



ARP Cache:

```
IP: 172.25.165.43  MAC: dc:0e:a1:a5:57:79
IP: 172.25.165.66  MAC: 14:da:e9:a9:05:9c
IP: 172.25.165.143  MAC: 20:6a:8a:4a:5f:b4
IP: 172.25.165.254  MAC: 00:24:14:30:6f:ff
```

☐ Autostart

☐ Terminate Wi-Fi if alerted

☑ Show status in notification bar

STATUS: RUNNING

Checks per (Minute): 60

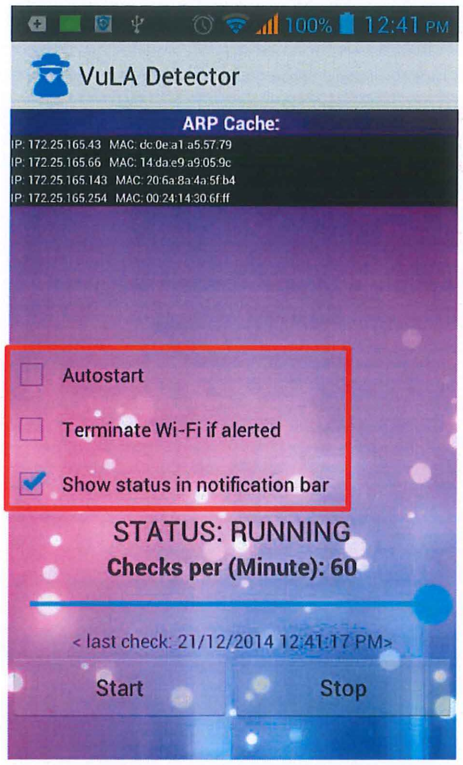
< last check: 21/12/2014 12:41:17 PM>

Figure 8 Main Interface

VuLA Detector is designed to able to provide response to the user. When the user launches the VuLA Detector Android application without connecting to the wireless network. The application is unable to execute and display the ARP information. However, the notify message is able ensure the user go check back their wireless connection either it is functioning well. Move over, if there is no any wireless connection of the devices, so the Start button is unable to press it. The interaction between the user and the application is fully performed well in the VuLA Detector Android application.

4.2.1 Main Interface's Checkboxes

These checkboxes are used for setting up the action would be done when the application is running. First checkbox is "Autostart" that means when the user activates the Wi-Fi connection, the application will start the detection process automatically. This function makes the smartphone be always safe anytime when connect to the internet. Second checkbox is "Terminate Wi-Fi if alert" that means when some circumstance of the attack is approaching into the smartphone, the alert will be show out and the Wi-Fi is disable automatically. This action is taken because to prevent the attack from getting access of the smartphone. And the last one is "Show status in the notification bar" that means it uses to display the status of the application either its running or not on behind of processing application. The status can be found on top of the application.



```
ARP Cache:
IP: 172.25.165.43   MAC: dc:0e:a1:a5:57:79
IP: 172.25.165.66   MAC: 14:da:e9:a9:05:9c
IP: 172.25.165.143   MAC: 20:6a:8a:4a:5f:b4
IP: 172.25.165.254   MAC: 00:24:14:30:6f:ff
```

[ ] Autostart

[ ] Terminate Wi-Fi if alerted

[✓] Show status in notification bar

STATUS: RUNNING

Checks per (Minute): 60

< last check: 21/12/2014 12:41:17 PM>

Figure 9 Main Interface's Checkboxes

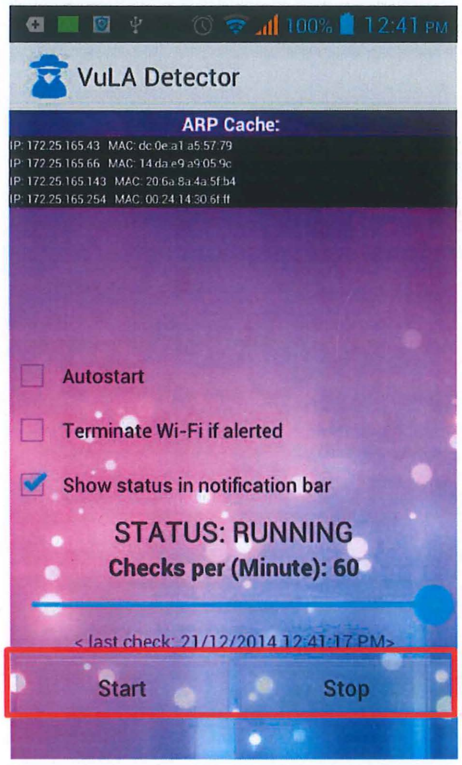
```
<CheckBox
Android:id="@id/autostart"
Android:layout_width="wrap_content"
Android:layout_height="wrap_content"
Android:checked="true"
Android:text="@string/start_when_wifi_enable"
Android:textSize="14.0dip"
Android:textColor="@Android:color/black" />
```

```
<CheckBox
Android:id="@id/autodisconnect"
Android:layout_width="wrap_content"
Android:layout_height="wrap_content"
Android:checked="true"
Android:text="@string/disconnect_wifi_if_alerted"
Android:textSize="14.0dip"
Android:textColor="@Android:color/black" />

 <CheckBox
Android:id="@id/notification"
Android:layout_width="wrap_content"
Android:layout_height="wrap_content"
Android:checked="true"
Android:text="@string/notification"
Android:textSize="14.0dip"
Android:textColor="@Android:color/black" />
```

4.2.2 Main Interface's Buttons

In the main interface, there are two buttons which are "Start" button and "Stop" button. When the start button is pressed, the application will be start the detection process of the ARP spoofing in the network that the smartphone connected. The status of the application will change into running stage. The list view will list out the available of all devices of the IP-MAC addresses in the ARP Cache. And when the stop button is pressed, the detection process will stop immediately and the ARP Cache will erase all mappings of IP-MAC from the list view.


100% 12:41 PM

VuLA Detector

ARP Cache:

```
IP: 172.25.165.43  MAC: dc:0e:a1:a5:57:79
IP: 172.25.165.66  MAC: 14:da:e9:a9:05:9c
IP: 172.25.165.143  MAC: 20:6a:8a:4a:5f:b4
IP: 172.25.165.254  MAC: 00:24:14:30:6f:ff
```

☐ Autostart

☐ Terminate Wi-Fi if alerted

☑ Show status in notification bar

STATUS: RUNNING

Checks per (Minute): 60

< last check: 21/12/2014 12:41:17 PM>

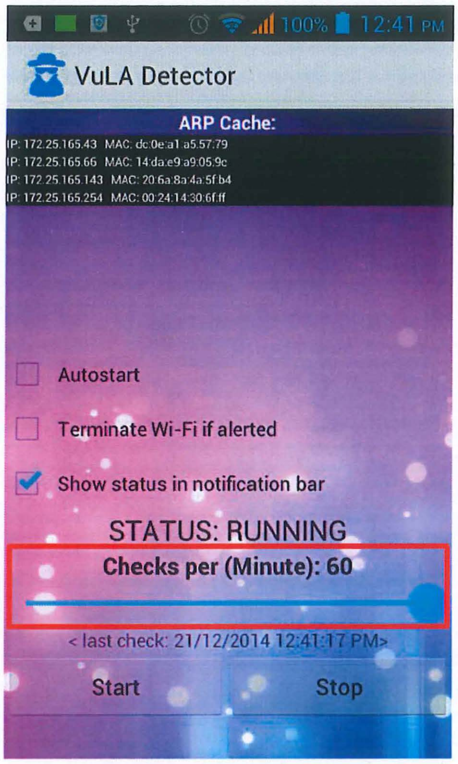
Figure 10 Main Interface's Buttons

```
<Button
Android:id="@id/btnStart"
Android:layout_width="wrap_content"
Android:layout_height="wrap_content"
Android:text="@string/start_monitoring"
Android:layout_weight="1.0"
Android:textColor="@Android:color/black" />

<Button
Android:id="@id/btnStop"
Android:layout_width="wrap_content"
Android:layout_height="wrap_content"
Android:text="@string/stop_monitoring"
Android:layout_weight="1.0"
Android:textColor="@Android:color/black" />
```

4.2.3 Main Interface's Seek Bar

The seek bar is one of the functions that the user can set the time interval for application to recheck the ARP Cache even the period of the mapping is not over. The user can select the time around of 60 minutes per check. So, this function makes this application was more special rather than other existing applications.



VuLA Detector

ARP Cache:

```
IP: 172.25.165.43  MAC: dc:0e:a1:a5:57:79
IP: 172.25.165.66  MAC: 14:da:e9:a9:05:9c
IP: 172.25.165.143  MAC: 20:6a:8a:4a:5f:b4
IP: 172.25.165.254  MAC: 00:24:14:30:6f:ff
```

- [ ] Autostart
- [ ] Terminate Wi-Fi if alerted
- [x] Show status in notification bar

STATUS: RUNNING

Checks per (Minute): 60

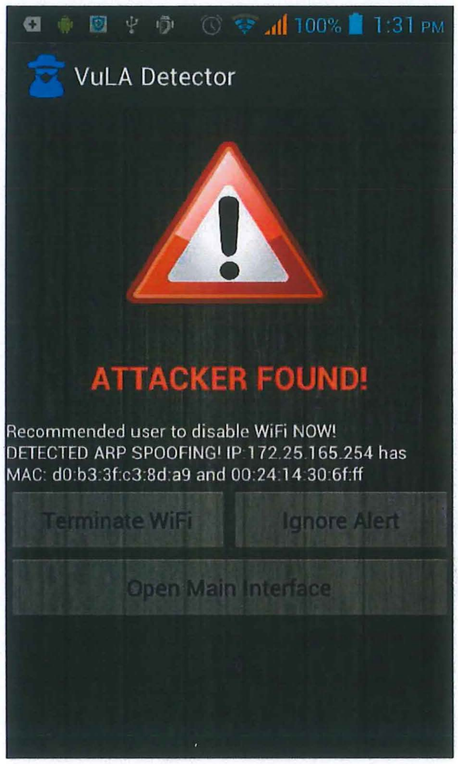
< last check: 21/12/2014 12:41:17 PM>

Figure 11 Main Interface's Seek Bar

```
<SeekBar
    Android:id="@id/checkinterval"
    Android:layout_width="match_parent"
    Android:layout_height="wrap_content"
    Android:max="60"
    Android:progress="1" />
```

4.3 ALERT INTERFACE

Alert interface is the second interface of the Android application. This interface will come out when the ARP spoofing is approached. The significance of this interface is it functions to show the attacker's information and the method to prevent the attack. There is a 1 text view and 3 buttons on alert interface.

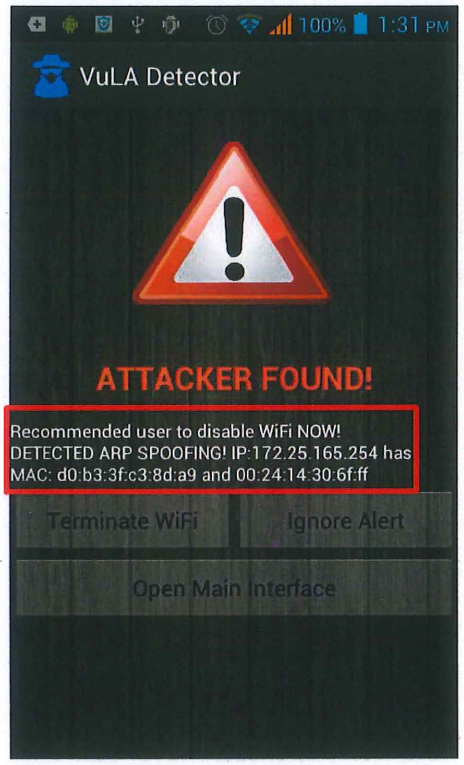


```
Recommended user to disable WiFi NOW!
DETECTED ARP SPOOFING! IP:172.25.165.254 has
MAC: d0:b3:3f:c3:8d:a9 and 00:24:14:30:6f:ff
```

Figure 12 Alert Interface

4.3.1 Alert Interface's Text View

The important information in alert interface is shown in text view. This info related to the ARP spoofing attacker. It shows the IP address that he uses to intrude to the network and have two different MAC addresses which map to one IP address. For details it looks like below:



Recommended user to disable WiFi NOW!
DETECTED ARP SPOOFING! IP:172.25.165.254 has
MAC: d0:b3:3f:c3:8d:a9 and 00:24:14:30:6f:ff

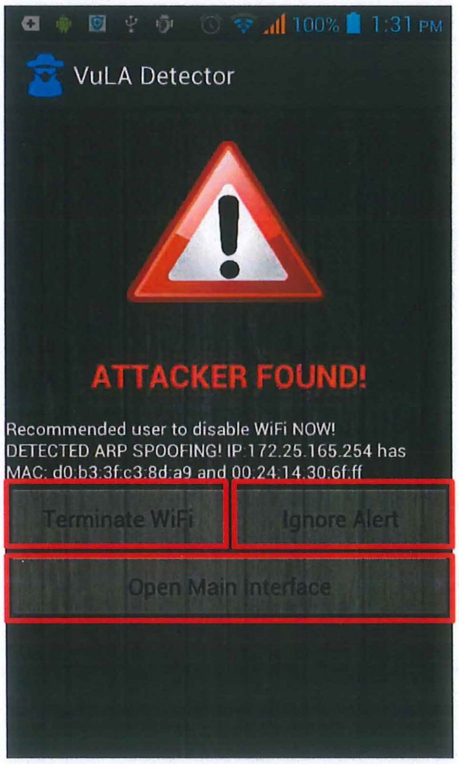
Figure 13 Alert Interface's Text View

```
<TextView
    Android:id="@id/disconnected"
    Android:layout_width="fill_parent"
    Android:layout_height="wrap_content"
    Android:layout_marginBottom="100dp"
    Android:text="Connection Status"
    Android:textSize="14.0sp"
```

Android:textColor="@Android:color/white" />

4.3.2 Alert Interface's Buttons

For the alert interface, it has 3 buttons that have functions related to the result of the detection process by took the action to the ARP spoofing attack. Firstly, "Terminate Wi-Fi" button can be used to terminate the Wi-Fi connection when the attack is already detected. This function provided because to make the smartphone to be secure and safe from the ARP spoofing attack. Secondly, "Ignore Alert" button which means that users don't want takes any action to the ARP spoofing and let it be in the smartphone. And then lastly, "Open Main Interface" that really clear its meaning that will be takes the user back to the first interface or called as the main interface.



ATTACKER FOUND!

Recommended user to disable WiFi NOW!
DETECTED ARP SPOOFING! IP:172.25.165.254 has
MAC: d0:b3:3f:c3:8d:a9 and 00:24:14:30:6f:ff

Figure 14 Alert Interface's Buttons

```
<Button
        Android:id="@id/btnDisconnectWifi"
        Android:layout_width="fill_parent"
        Android:layout_height="wrap_content"
        Android:layout_weight="1.0"
        Android:text="Terminate WiFi" />

<Button
        Android:id="@id/ignore"
        Android:layout_width="fill_parent"
        Android:layout_height="wrap_content"
        Android:text="@string/ignore"
        Android:layout_weight="1.0" />

<Button
     Android:id="@id/openAdmin"
     Android:layout_width="fill_parent"
     Android:layout_height="wrap_content"
     Android:text="@string/open_main" />
```

4.4 DETECTION ALGORITHM AGAINST ARP SPOOFING

The algorithm of detection of ARP spoofing in the same network is started when the application collects all devices' IP addresses and MAC addresses which send an ARP request to the user's device in the same network include the router. If the IP address is same as the IP address of the router, but if the IP address is varying as the IP address of a router, then it will be ignored. After the first phase is done, the second phase continues by comparing the router's MAC address with other MAC address which has the same IP address to the router to test whether its IP address contain two MAC addresses or not. The ARP Cache will always up to date. This is because all of ARP packet will be sold or put into the ARP Cache evens the mapping of that IP-MAC address is already done. When two MAC addresses in a router are mapped to an IP address then the collision was happening. And this is one of signs that someone is the network trying to spoof the ARP inside the network. After the detection is done, the application will switch to another interface which is an alert interface to alert user that something wrong inside the network.

```
while(true) {
  String var3 = var1.readLine();
  if(var3 == null) {
    var1.close();
    if(this.warnUser) {
      this.warnUser(var2);
    }

    if(handler != null) {
      Message var15 = handler.obtainMessage();
      var15.setData(new Bundle());
      var15.getData().putBoolean("UPDATE", true);
    var15.getData().putSerializable("ARPTABLE", this.arpTableList);
    handler.sendMessage(var15);
  }

  return;
}

if(!var3.startsWith("IP")) {
  String[] var4 = var3.split("[ ]+");
  String var5 = var4[0];
  String var6 = var4[3];
  Log.d("VuLA Detector", "IP: " + var4[0] + " MAC: " + var4[3]);
  if(!checkMapARP.containsKey(var5)) {
    checkMapARP.put(var5, var6);
   } else if(!var6.equalsIgnoreCase((String)checkMapARP.get(var5)) &&
  !var6.equalsIgnoreCase("00:00:00:00:00:00") &&
  !var6.equalsIgnoreCase("FF:FF:FF:FF:FF:FF")) {
    var2 = "DETECTED ARP SPOOFING! IP:" + var5 + " has MAC: " + var6 + "
  and " + (String)checkMapARP.get(var5);
    if(var5.equalsIgnoreCase(this.gateway)) {
      this.warnUser = true;
    }

    Log.d("VuLA Detector", var2);
    Iterator var9 = checkMapARP.keySet().iterator();
```

```
          while(var9.hasNext()) {
            String var10 = (String)var9.next();
            Log.d("VuLA Detector", "  > " + var10 + " = " +
(String)checkMapARP.get(var10));
          }
        }

        String var12 = "IP: " + var5 + "   MAC: " + var6;
        if(this.arpTableList.indexOf(var12) < 0) {
          this.arpTableList.add(var12);
        }
      }
    }
```

4.5 CONDITIONS AND FORMULA FOR DETECTION OF ARP SPOOFING

The Address Resolution Protocol (ARP) is a stateless protocol which means it does not has authentication mechanism for identifying and verifying the identity of the sender because of that it has a long history of being prone to spoofing attack. For this research, the active technique is chosen as a solution to defense from ARP spoofing attack because it is the best technique compared to use the passive technique. There are the normal and abnormal conditions that could be found during the detection process in the ARP cache.

Table 5 Types of Conditions in ARP Cache

Normal Condition	Abnormal Condition
A = IP(A) + MAC(A)	**A = IP(A) + MAC(A)**
B = IP(B) + MAC(B)	**B = IP(B) + MAC(B)**
C = IP(C) + MAC(C)	**C = IP(B) + MAC(C)**

The algorithm of the detection process is started with comparison between router's IP address and IP addresses of users. Then, comparison between router's MAC address and MAC addresses of users. If a router's IP address is not same with the IP addresses of users, then save their IP and MAC address and continue monitoring process. But if the router's IP address is same with the IP addresses of users, then the MAC address of users which not same with the router's MAC address is declared as a suspicious person in the network. The formula for detection process is illustrated as below:

```
(ROUTER) A = IP(A) + MAC(A)                   } COMPARE
(USERS) B/C = IP(B)/(C) + MAC(B)/(C)          }

If  IP(B)/(C)  !=  IP(A), then
Save IP and MAC (B) /(C) and continue monitoring…
Else If  IP(B)/(C)  =  IP(A), then
Any MAC(B)/(C)  !=  MAC(A)/(Broadcast Address)
DETECTED
```

Figure 15 The Algorithm of Detection Process

4.6 PROTECTION TECHNIQUE AGAINST ARP SPOOFING

For the protection technique against ARP spoofing in the network is just a very simple and the safest way which is terminated the Android user's Wi-Fi connection once ARP spoofing is approached in the network. The user can choose auto termination of Wi-Fi or manually terminate the Wi-Fi by unchecking the checkbox of auto termination. If for the other technique using BSSID is not 100 percent accurate and user also needs root their Android device which is illegal to many of company who making Android devices.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 INTRODUCTION

In this chapter, the result of the VuLA Detector Android application after completion of the design and implementation will be conducted. Not only that, the hardware device like TP-Link wireless access point has setting up and functioning well in the testing phase. The testing phase is to evaluate the components, use of VuLA Detector Android application and the application performance and reliability.

Therefore, all the results and output will be discussed from the testing phase. Besides that, the result analysis and the research, constraint will be conducted and discussed in this chapter. Based on the result and discussion, it will help to identify the limitations and the strengths of the VuLA Detector Android application as well as the future enhancements on the application.

5.2 RESULT ANALYSIS

After the completion and development of the VuLA Detector Android application. It has met all the objectives of the project, which are:

i. To study the vulnerability of ARP spoofing attack on a private wireless network.
ii. To apply the active technique on detection of ARP spoofing attack for Android devices.
iii. To develop ARP spoofing detection prototype for Android users who connect their Android devices on a private wireless network.

The VuLA Detector Android application has met the first objective of the research by studying the ARP spoofing on private wireless network and knows how to handle this problem by downloading the ARP spoofing application from the third party marketing. Based on the research, those ARP spoofing application does not require commands or hacking skill to run it on the network. The normal user can be a hacker because it just launch the attack by pressing one button because the application has the GUI which very easy for newbies to use it. Therefore, the application against the ARP spoofing needs to develop for the Android operating system. Many applications were developed to detect and protect ARP spoofing today and almost of them are for Windows OS platform. The iOS Apple is not an open source OS, which cannot make any change on it. The Android OS is selected because it is an open source OS and nowadays many users around the world using Android as their smartphone platform.

The VuLA Detector Android application has met the second objective of the research by studying about a few existing techniques to detect and protect against ARP spoofing. All the techniques are usable and effective. But in Android OS there is a limitation which not all the techniques are suitable use in the Android OS. One of the techniques of detection used in the application is to create a log file that contains all devices IP addresses and MAC addresses. Then use the IP address and MAC address of the router as a main to compare with others IP addresses and MAC addresses. If someone tries to use ARP spoofing, there will be a collision in the ARP Cache. And for the protection technique, the best solution to prevent an ARP spoofing attack the smartphone by terminating the Wi-Fi connection. This technique may look simple, but it is a safe way and legal for some company compared to the technique of immune that requires a rooted smartphone to run it.

For the lastly objective of the research which has met by creating a prototype Android application that provides more security to Android user devices. The important information about the user will keep safe from lack out to hackers or attacker easily. The users can use their Android smartphone assured in the private home network. If something wrong is occurring, the most recommended solution of users is jump out from the private network and do not connect it again. The application will upload in the Google Play Store for free version and users can download it even anywhere they are.

5.3 CONSTRAINTS

The constraints of this research can be categorized into two parts:

i. Development Constraints
ii. System Constraints

5.3.1 Development Constraints

To codes an application needs to know how to do. Because this is the first time use the Eclipse software which is not familiar about Android, Java coding and something about it. Therefore, start learns from beginning all the basic Java coding for Android from website like www.youtube.com and www.Android.com. Some coding of the application is referred from other open source code. Then, find a suitable technique which that Linux-base OS can be used. In concluding, there is no perfect technique that can be used for detection and protection against ARP spoofing because all those require root access for the Android devices.

5.3.2 System Constraints

The application can be installed for Android devices only and not for Windows Mobile, iOS, Blackberry or Symbian. The immune technique that has found for Android device needs to root access using the Superuser. The Superuser application will give permission to read and write to the system files. The problem here is root access to Android device is illegal and the device's warranty will void instantly. The detection application against ARP spoofing is only for users inside the same network. If the hacker comes from the different network then this application does not work.

5.4 ENHANCEMENT FOR FUTURE RESEARCH

There are a few enhancements that can be carried out for future improvement of the application:

i. The perfect protection technique against ARP spoofing will be applied in the application, but require the root access for sure. Through this technique, the user can still use the internet although ARP spoofing still playing in the network.
ii. Set the alarm sound notification to the application. The user can choose whatever types of sound or music to be used as its notification sound.
iii. For the future application development, it is focus on to develop an mobile application which is can support for multiple mobile operating system such as iOS, Windows Phone, Blackberry, and etc.

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

In conclusion, the research on detection of vulnerability attack is focused on how to detect the ARP spoofing attack on the Android platform. The research trip runs smoothly and successfully. During this research, so much has been learned and understands what is ARP spoofing and how quite dangerous it is and when the attacker uses it on the private network. Besides, many people do not know what spoofing is and how to protect from it. This is because they just know to connect their smartphone to the private network and feel safe alone without realizing that they are being watched by an attacker. In addition, there are less Android applications that provide detection and protection function against ARP spoofing in Google Play Store, some need to pay and some is not functioning as well.

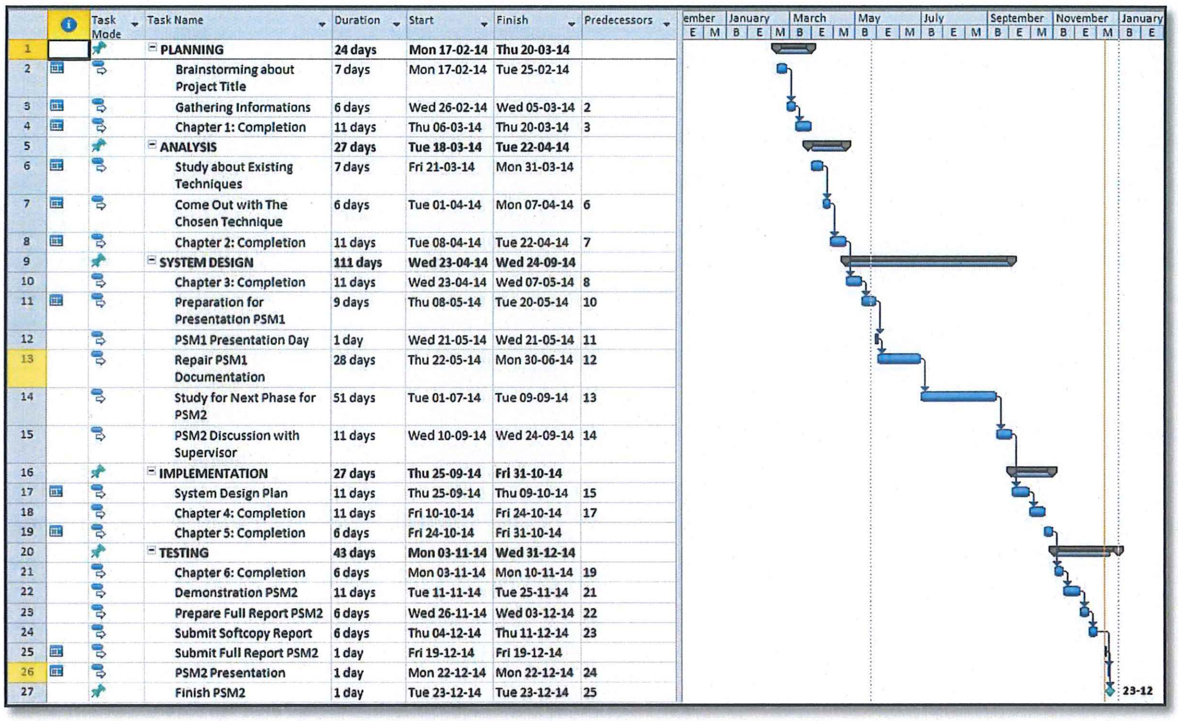
Therefore, the prototype of the application is developed and finally done. It can run smoothly on Android emulator and real Android devices. The technique that uses for applying on the Android application is not the best technique, but it can function very well provide the detection and protection against ARP spoofing awhile connecting to the private wireless network. It is free to download on Google Play Stores and use for fully functional that it is already stable enough for the user to use it. The application is done through a lot of constraints during the development of the application and system itself. All the tasks along the research is followed exactly over time and schedule based on the Gantt chart. In future, the advance technique will be used to improve the level of protection and its performance to get better than this. Besides, the available version for Android will be upgraded from time to time in order the application can be used by all Android users. For the last word, the application will be developed for another mobile OS.

REFERENCES

1) Ramachandran, V., and Nandi, S. (2005). Detecting ARP Spoofing: An Active Technique, LNCS Publication, Proceedings of the 1st International Conference on Information Security Systems, pp 239-250.
2) Kumar, V. (2012). Introduction to ARP-Poisoning. Retrieved October 10,2012,from http://basichackingskills.wordpress.com/2012/02/25/introduction-to-arp-poisoning/
3) Barra, H. (2012). 500 MILLION. Retrieved October 8, 2012, from https://plus.google.com/u/0/110023707389740934545/posts/R5YdRRyeTHM
4) Wikipedia (n.d.). ARP spoofing. Retrieved October 10, 2012, from http://en.wikipedia.org/wiki/ARP_spoofing
5) Rouse, M. (2006). Session Hijacking (TCP session hijacking). Retrieved October 18, 2012, from http://searchsoftwarequality.techtarget.com/definition/session-hijacking
6) Lane H. D. (2003). Security Vulnerabilities and Wireless LAN Technology, from http://www.sans.org/int/whitepapers/wireless/1629.php. (4 November 2012).
7) Nikiforakis, N., Meert, W., Younan, Y., Johns, M., and Joosen, W. (2011). SessionShield: Lightweight Protection against Session Hijacking. Engineering Secure Software and Systems, pp 87-100.
8) Liu, Y., Dong, K., Dong, L., and Li, B. (2008). Research of the ARP Spoofing Principle and a Defensive Algorithm. WSEAS TRANSACTIONS on COMMUNICATIONS, pp 518-519.
9) Gill, Rupinder S. and Smith, Jason and Looi, Mark H. and Clark, Andrew J. (2005). Passive techniques for detecting session hijacking attacks in IEEE 802.11 wireless networks, 4-5.

APPENDIX A

GANTT CHART



		Task Mode	Task Name	Duration	Start	Finish	Predecessors
1			**PLANNING**	**24 days**	**Mon 17-02-14**	**Thu 20-03-14**	
2			Brainstorming about Project Title	7 days	Mon 17-02-14	Tue 25-02-14	
3			Gathering Informations	6 days	Wed 26-02-14	Wed 05-03-14	2
4			Chapter 1: Completion	11 days	Thu 06-03-14	Thu 20-03-14	3
5			**ANALYSIS**	**27 days**	**Tue 18-03-14**	**Tue 22-04-14**	
6			Study about Existing Techniques	7 days	Fri 21-03-14	Mon 31-03-14	
7			Come Out with The Chosen Technique	6 days	Tue 01-04-14	Mon 07-04-14	6
8			Chapter 2: Completion	11 days	Tue 08-04-14	Tue 22-04-14	7
9			**SYSTEM DESIGN**	**111 days**	**Wed 23-04-14**	**Wed 24-09-14**	
10			Chapter 3: Completion	11 days	Wed 23-04-14	Wed 07-05-14	8
11			Preparation for Presentation PSM1	9 days	Thu 08-05-14	Tue 20-05-14	10
12			PSM1 Presentation Day	1 day	Wed 21-05-14	Wed 21-05-14	11
13			Repair PSM1 Documentation	28 days	Thu 22-05-14	Mon 30-06-14	12
14			Study for Next Phase for PSM2	51 days	Tue 01-07-14	Tue 09-09-14	13
15			PSM2 Discussion with Supervisor	11 days	Wed 10-09-14	Wed 24-09-14	14
16			**IMPLEMENTATION**	**27 days**	**Thu 25-09-14**	**Fri 31-10-14**	
17			System Design Plan	11 days	Thu 25-09-14	Thu 09-10-14	15
18			Chapter 4: Completion	11 days	Fri 10-10-14	Fri 24-10-14	17
19			Chapter 5: Completion	6 days	Fri 24-10-14	Fri 31-10-14	
20			**TESTING**	**43 days**	**Mon 03-11-14**	**Wed 31-12-14**	
21			Chapter 6: Completion	6 days	Mon 03-11-14	Mon 10-11-14	19
22			Demonstration PSM2	11 days	Tue 11-11-14	Tue 25-11-14	21
23			Prepare Full Report PSM2	6 days	Wed 26-11-14	Wed 03-12-14	22
24			Submit Softcopy Report	6 days	Thu 04-12-14	Thu 11-12-14	23
25			Submit Full Report PSM2	1 day	Fri 19-12-14	Fri 19-12-14	
26			PSM2 Presentation	1 day	Mon 22-12-14	Mon 22-12-14	24
27			Finish PSM2	1 day	Tue 23-12-14	Tue 23-12-14	25

Figure 16 The Gantt Chart of Workflow