DETECTION OF MAN-IN-THE-MIDDLE ATTACK IN IEEE 802.11 NETWORKS

BY

SALIFU, ABDUL-MUMIN, BSc. Computer Science (Hons.)

A Thesis Submitted To The Department Of Electrical and Electronic

Engineering,

Kwame Nkrumah University Of Science And Technology

In Partial Fulfillment Of The Requirements For The Degree

Of

MASTER OF SCIENCE IN TELECOMMUNICATION ENGINEERING

MAY, 2011

Declaration

I, Salifu Abdul-Mumin, hereby declare that this thesis, "Detection of Man-the-Middle Attack in IEEE 802.11 Networks", consists entirely of my own work produced from research undertaken under supervision and that no part of it has been published or presented for another degree elsewhere, except for the permissible references from other sources, which have been duly acknowledged.

Date…….. Sgd……

(Salifu Abdul-Mumin)

Date ……. Sgd……..

(Supervisor: Dr. K. O. Boateng)

Date……. Sgd……

(Head of Department)

Dedication

This work is dedicated to my late father, Mr. Salifu Demah Selanwiah for his fatherly responsibility, love, care and encouragement. May the Good Lord grant him one of the places in paradise.

Acknowledgement

"Silver and Gold I have none but such as this I give unto you". The success of this project is a result of the combined effort and contributions of many people who in one way or other deserved to be acknowledged.

I wish to express my sincerest thanks to the Almighty God for his protection and guidance, which has taken me to this height.

I acknowledge with a sentiment of deep gratitude to my supervisor Dr. K. O. Boateng for his relentless efforts, patience and overwhelming understanding in guiding and offering me constructive comments, directions and useful suggestions that made this work possible.

Many thanks also go to all lecturers of the faculty of Electrical and Electronic Engineering who natured me in one way or the other and has contributed significantly to the successful completion of my degree.

I further express my thanks to my family who gave me prayers and constructive advice, which has seen me this far.

To my friends and course mates, I am grateful and wish to thank you for the manifold support and encouragement.

Abstract

The recent proliferation of high speed wireless portable devices has resulted in a significant demand for wireless LANs. WLANs offer tremendous flexibility to the mobile user, compared with their wired LANs equivalent. With this development, Wireless networks are vulnerable to many identity-based attacks in which a malicious device uses forged MAC addresses to masquerade as a specific client. One of these attacks is Man-in-the-middle attack.

Interestingly the man-in-the-middle is usually part of the network where malicious activities are mostly least expected. Most stakeholders turn to secure their networks against unauthorized users paying less attention to those within their own networks.

In solving this malicious activity, we studied the state of the act of executing man-in-the-middle attack in wireless networks. Based on this study, a mechanism has been developed to detect this attack. We generated a list of database consisting of IP addresses and MAC addresses of authorized users on a WLAN to check on packets coming to the destination host. The detection mechanism has been simulated in a wireless environment to test the effectiveness of the detection mechanism. In all cases the simulated man-in-the-middle attack was detected.

The performance of the proposed detection mechanism was evaluated against an existing detection mechanism.

Table of Contents

LISTS OF TABLES

LISTS OF FIGURES

ACRONYMS

MITM: Man-in-the-middle attack

TCP: Transmission Control Protocol

DNS: Domain Name System

IP: Internet Protocol

ARP: Address Resolution Protocol

FTP: File Transmission Protocol

SSL: Secure Sockets Layer

DHCP: Dynamic Host Configuration Protocol

LANs: Local Area Networks

WLANs: Wireless Local Area Networks

MAC: Media Access Control

DoS: Denial of service attack

IEEE: Institute of Electrical and Electronic Engineers

WEP: Wired Equivalent Privacy

FBI: Federal Bureau of Investigation

RSNA: Robust Security Network Association

TKIP: Temporal Key Integrity Protocol

CCMP: Counter-mode/CBC-MAC Protocol

TLS: Transport Layer Security

EAP: Extensible Authentication Protocol

TKs: Temporal keys

RADIUS: Remote Authentication Dial in User Service

NIC: Network Interface Card

CRC: Cyclic Redundancy Checksum

AP: Access Point

SSID: Service Set ID

BSS: Basic Service Set

TCP: Transmission Control Protocol

TCP/IP: Transmission Control Protocol/Internet Protocol

FTP: File Transmission Protocol

UDP: User Datagram Protocol

HTTP: Hyper Text Transfer Protocol

WWW: World Wide Web

URL: Uniform Resource Locator

ICMP: Internet Control Message Protocol

OS: Operating System

IDS: Intrusion Detection Systems

WIDSs: Wireless Intrusion Detection Systems

RSS: Received Signal Strength

RSSI: Received Signal Strength Indication

RF: Radio Frequency

NAV: Network Allocation Vector

RTS: Request to Send

CTS: Clear to Send

PC: Personal Computer

DES: Data Encryption Standard

IBM: International Business Machines

RSA: Rivest, Shamir and Adleman

Chapter 1

Introduction

1.0.0 Background

Wireless Threats: From the Link Layer of a WLAN, there are three possible types of frames: Management Frames, Control Frames, and Data Frames.[1] Any manipulation of these frames that directly or potentially jeopardizes data confidentiality, integrity, mutual authentication, and availability will be considered a threat. [1]

The types of security threats faced by wireless networks are many and varied, and although initially targeted at the Physical (PHY) and Media Access Control (MAC) layers, the ultimate goal is to access or disrupt data or activity at the application layer.[2] Some of these vulnerabilities are Denial of service (DoS) attack, Jamming, Insertion attacks, Replay attack, ARP Spoofing, Man-in-the-Middle attack, Cryptanalytic attacks etc.

Although the range of threats is wide and varied, in most cases executing these kinds of attack requires a high level of technical expertise on the part of the hacker.[3]

1.1.0 Man-in-the-middle attack

Man-in-the-middle attack (often abbreviated **MITM**), is a form of active eavesdropping in which the attacker makes independent connections with the victims and relays messages between them, making them believe that they are talking directly to each other over a private connection when in fact the entire conversation is controlled by the attacker. A man-in-the-middle attack can only be successful when the attacker can impersonate each endpoint to the satisfaction of the other. [4]. MITM is also known as:

- ✓ Bucket-brigade attack

- ✓ Fire brigade attack
- ✓ Monkey-in-the-middle attack
- ✓ Session hijacking
- ✓ TCP hijacking

1.1.1 Origin of name

The name “Man-in-the-middle” is derived from the basketball scenario where two players intend to pass a ball to each other while one player between them tries to seize it. MITM are sometimes referred to as “bucket brigade attacks” or “fire brigade attacks”. Those names are derived from the fire brigade operation of dousing off the fire by passing buckets from one person to another between the water source and the fire. [5]

1.1.2 Other definitions

MITM has been defined as a type of attack where a user gets between the sender and receiver of information and sniffs any information being sent. [6]

Another author has defined MITM as attacks in which the attacker infiltrates unnoticed the communication channel between two partners and is thereby able to spy on or even modify their data exchanges. [6]

Man-in-the-Middle attack is the type of attack where attackers intrude into an existing connection to intercept the exchanged data and inject false information. It involves eavesdropping on a connection, intruding into a connection, intercepting messages, and selectively modifying data. [7]. MITM attacks are often referred to as “session hijacking attacks”, suggesting that the intruder aims to gain access to a legitimate user’s session to tamper

it. The attacker usually starts with sniffing and eavesdropping on a network stream, and ends with trying to alter, forge or reroute the intercepted data.[8] One of the objectives for MITM attacks is to gain access to the client's messages and modify them before finally transmitting them to the server end. Other objectives of MITM can be to mislead the communicators at the client or server end, to intercept pertinent information (e.g., identity, address, password, or any other confidential information for malicious purposes) and also, at times, manipulate transactions.

1.1.3 Scenario

In the real world game of keep-away, two people toss a ball back and forth while a third person – the man in the middle – tries to intercept the ball while it is enroute. In the cyber world, the game of keep-away gets a new twist; the two players have no idea the man in the middle (MITM) exists. It works like this:

- ✓ Computer A initiates conversation with computer B
- ✓ Computer C intercepts the attempt and then relays the request to computer B
- ✓ Computer B responds, computer C intercepts it, and returns that response to computer A.

While computer C has the intercepted communication, it can modify the communication or even redirect it to an entirely new destination (i.e. Computer D). Meanwhile, computer A continues to believe that it is communicating only with computer B.

Computer C has been able to interject itself between A and B through a process known as ARP poisoning**. [**9**]**

DNS poisoning is another form of MITM attack. The DNS, or Domain Name System, resolves IP addresses to domain names. Vulnerabilities on the DNS server can allow attackers to insert malicious DNS information, for example directing all attempts to access a particular banking site to a lookalike site under the attacker's control.

Hosts file manipulation is another method used to redirect traffic. Every Windows-based computer has a local Hosts file which, like DNS, resolves IP address to domain names. However, entries in the local Hosts file typically override DNS and the Hosts file is generally more accessible to attackers – thus malicious Hosts file manipulation is common. [9]

1.2 Problem Statement

With the business of e-commerce on its peak, more sensitive information is being passed around on computer networks. Financial and identity information are at a higher risk of being stolen or modified as users take advantage of the ease of doing business online through web applications. Sensitive user information is constantly transported between sessions after authentication and hackers are putting their best efforts to steal or modify them.[10]

Computer networks face a variety of serious threats and risks. These threats are based on vulnerabilities associated with the ARP.[9] ARP, or Address Resolution Protocol, uses a 'pick me' approach to resolving computers on a network. When computer A tries to communicate with B, ARP sends out a broadcast to the network devices asking 'who is B? But there is no authentication built into ARP [9] and thus ARP has no way of determining whether the response (pick me) is really B or not. By exploiting this lack of authentication, a malicious computer can

tell ARP it is computer B, after which ARP will begin directing future requests for computer B to the malicious computer. [9].

The final consequence is the disclosure of data which can be an act of economic terrorism, alteration of data such as grade fixing and denial-of-service attacks including Synchronization (SYN) floods and smurfing.

1.3 Objectives of the Study

The general objective of this thesis is to contribute towards the maintenance of secure network systems.

Objectives to this study are:

Specific objectives are:

1. To propose an efficient passive technique for detecting Man-in-the-Middle attacks in computer networks.
2. To design and implement computer program based on the proposed technique for simulation purposes.

1.4 Scope of Study

This research involves using the state of the art of MITM attack to propose and simulate a protocol that will be used in detecting MITM. We will be illustrating how to execute MITM attacks in IEEE 802.11 network. We will also illustrate and analyze how to detect this problem and consequently proposed an efficient passive detection mechanism that can be used to address this attack. Based on the detection mechanism, a computer program will be developed for simulation purposes to ascertain the effectiveness of the proposed detection mechanism. We will

further evaluate the performance of the proposed detection mechanism against an existing detection method that will be discussed in the next chapter.

1.5 Justification

MITM is an exploit that targets the victims TCP based applications like Telnet, rlogin, ftp, mail application, web browser etc. An attacker can grab unencrypted confidential information from a victim's network base TCP application. He can further tamper the Authenticity and Integrity of the data. A security researcher by name Moxie Marlinspike has demonstrated a way to hijack

Secure Sockets Layer (SSL) sessions to intercept login data. The anarchist researcher explained in a YouTube video that the attack uses a tool developed called SSLstrip, which exploits the interface between http and https sessions. In his Black Hat presentation, he claimed to have gathered details on 117 email accounts, seven PayPal logins and 16 credit card numbers, within a 24 hour period. It is therefore prudent to have an efficient detection mechanism that will always flag out these activities on wireless network.

1.6 Assumptions

There is no such thing as perfect security; we need to consider how a system will react to a successful attack. Indeed the most critical part of a security system is not how well it works but how well it fails. In this theses work, the following assumptions are made.

1. The strategy to be developed is not to be used for a network on which a Dynamic Host Configuration Protocol (DHCP) server is enabled.
2. The strategy is to capture packets online and apply offline methods to analyze the captured packets.

1.7 Methodology

The main objective of the research is to write a computer program that can be used to detect MITM. This is to be done by studying the state of the art of executing MITM and coming out with an effective detection protocol. Based on the study outcome, software will be designed to simulate the protocol. Finally a simulation will be conducted to verify the effectiveness of the strategy. We will conduct an evaluation against an existing detection method.

1.8 Organization of Thesis

This chapter provides an overview of the research undertaken in this thesis. The remainder of the work is organized as follows:

Chapter two, titled “Review of Literature”, summarized works that have been done in the area of Computer networks, Network Security, Man-in-the-Middle attack, and ARP Spoofing.

Chapter three, “Software Design”, gives a detail description of the use of the state of the act of executing Man-in-the-Middle attack on computer networks and how we can use this knowledge to detect this malicious act. The chapter further presents how the software was designed, using ActivePerl 5.101.

The next chapter is chapter four. It is titled “Implementation and Testing”, which presents the experimental setup and how the software was used to simulate the proposed protocol.

The thesis is completed with conclusion and recommendations which is presented in chapter five.

Chapter 2

Review of Literature

2.0.0 Wireless Security

The recent proliferation of high speed wireless portable devices has resulted in a significant demand for wireless LANs.[1] While WLANs offer tremendous flexibility to the mobile user, compared with their wire-line equivalent, WLANs are more susceptible to security threats because data is transmitted through the air**.** Wireless security is the prevention of unauthorized access or damage to computers using wireless networks.[4] Wireless networks are very common; both for organizations and individuals. Many laptop computers have wireless cards pre-installed.

The ability to enter a network while mobile has great benefits. However, wireless networking has many security issues. Hackers have found wireless networks relatively easy to break into, and even use wireless technology to crack into wired networks.[5] As a result, it's very important that enterprises define effective wireless security policies that guard against unauthorized access to important resources. Wireless network Security has been under extensive scrutiny in the last few years and has received due attention from both academic and industrial sectors [4]. However despite the emergence of security standards like IEEE 802.11i that improve the confidentiality and integrity protection of user data, wireless networks still face a variety of serious threats and risk [4].

2.1.0Wireless Network Attacks

These are various attacks that are being executed in order to gain access or cause damage to computers using wireless networks. The types of security attacks faced by wireless networks are many and varied, and although initially targeted at the Physical (PHY) and Media Access

Control (MAC) layers, the ultimate goal is to access or disrupt data or activity at the application layer.[2]

2.1.1 Passive Eavesdropping/Traffic Analysis

Due to the characteristics of wireless communication, an adversary can easily sniff and store all the traffic in a WLAN. Even when messages are encrypted, it is important to consider whether an adversary may learn partial or complete information from certain messages. This possibility exists if common message fields are predictable or redundant; further, encrypted messages may be generated upon the requests from the adversary itself.[20]

2.1.2 Message Injection/Active Eavesdropping

An adversary is capable of inserting a message into the wireless network with moderate equipment, such as a station with a common wireless Network Interface Card (NIC) and some relevant software.[11] Although the firmware of most wireless NICs may limit the interface for composing packets to the 802.11 standard, an adversary is still able to control any field of a packet using known techniques. [11] This assumes that an adversary can generate any chosen packet, modify contents of a packet, and completely control the transmission of the packet. The adversary can also insert a replayed packet, if there is no replay protection or the adversary is able to avoid it. Furthermore, by inserting some well-chosen packets, the adversary might be able to learn more information from the reaction of the system through active eavesdropping.

2.1.3 Message Deletion and Interception

An adversary is capable of removing a packet from the network before it reaches its destination. [12]This could be done by interfering with the packet reception process on the receiver's antenna, for example by causing CRC (Cyclic Redundancy Checksum) errors so that the receiver drops the packet. This process is similar to ordinary packet errors due to noise, but may be

instigated by an adversary. Message interception means that an adversary is able to control a connection completely. In other words, the adversary can capture a packet before the receiver actually receives it, and decide whether to delete the packet or forward it to the receiver. This is more dangerous than the eavesdropping and message deletion. [12]

This attack differs from eavesdropping and replaying, because the receiver does not get the packet before the adversary forwards it. Message interception may seem difficult in wireless LANs because the legitimate receiver might detect a message as soon as the adversary does so.

However, a determined adversary does have some potential ways to achieve message interception. [12] For example, the adversary can use a directional antenna to delete a packet on the receiver side, while simultaneously using another antenna to receive the packet itself.

2.1.4 Masquerading and Malicious AP

Because the plaintext MAC addresses are included in all packets transmitted through wireless links, an adversary can learn valid MAC addresses by eavesdropping. The adversary is also capable of modifying its MAC address to any value because most firmware provides the interface to do so.[13] If a system uses MAC address as the only identification of the wireless devices, the adversary can masquerade as any wireless station by spoofing its MAC address; or can masquerade as an Access Point (AP) by spoofing its MAC address and functioning appropriately through appropriate freeware (e.g., HostAP). An adversary is also able to install his own AP, with a forged MAC address and a spoofed SSID. Alternatively, without masquerading as others, it is possible for a malicious AP to provide a strong signal and attempt to fool a wireless station into associating with it and leaking credentials or private data.[13]

2.1.5 Denial-of-Service

WLAN systems are quite vulnerable to DoS attacks. An adversary is capable of making the whole Basic Service Set (BSS) unavailable, or disrupting the connection between legitimate peers.[14] Using characteristics of wireless networking, an adversary may launch DoS attacks in several ways. For example, forging the unprotected management frames (e.g., Deauthentication and Disassociation), exploiting some protocol weaknesses, or straightforward jamming of the frequency band will deny service to legitimate users.

2.1.6 Jamming

A jammer is a device which can partially or entirely disrupt a node's signal, by increasing its power spectral density (PSD). Jammer can never re-produce a signal nor can it pretend like a receiver node.[15] The parameters such as signal strength of a jammer, the location and the type influences the performance of the network and each jammer has different effect on the node..

2.1.7 Session Hijacking

Session hijacking is "a security attack on a user session over a protected network."[16] It involves employing various techniques to tamper with or take over TCP and web application user sessions. TCP session hijacking is when a hacker takes over a TCP session between two machines. Since most authentications only occur at the start of a TCP session, this allows the hacker to gain access to a machine. A popular method is using source-routed IP packets.[16] This allows a hacker at point A on the network to participate in a conversation between B and C by encouraging the IP packets to pass through its machine. If source-routing is turned off, the hacker can use "blind" hijacking, whereby it guesses the responses of the two machines. Thus, the hacker can send a command, but can never see the response. A hacker can also be "inline" between B and C using a sniffing program to watch the conversation. A common component of

such an attack is to execute a denial-of-service (DoS) attack against one end-point to stop it from responding. This attack can be either against the machine to force it to crash, or against the network connection to force heavy packet loss. If the session hijacker successfully impersonates the user/client, he gains access to the sensitive information found in the session [11].

Session hijack attacks are usually waged against users that are members of large networks containing a substantial number of open sessions. Network protocols like FTP, Telnet, and rlogin are especially attractive to the attacker, because of the session oriented nature of their connections, and the length of their communication sessions. Additionally, FTP, TELNET, and rlogin do not implement any security during logon, authentication, or data transmission. In fact, data sent using these protocols are sent in clear text which can be easily be viewed by anyone monitoring the network. [8]

Session hijacking combines denial of service and identity spoofing attacks.[17] Typically, an adversary forces a legitimate STA (Workstation) to terminate its connection to an AP (Access point) by sending it a disassociation/deauthentication management frame with the source MAC address spoofed to be that of the AP. This results in the STA disconnecting from the network. The adversary can now associate with the AP, by masquerading the MAC address of the STA, and hence taking over its session. The adversary can periodically spoof management frames to the legitimate STA to stop it from even retrying association with the AP. [17].

There are three different types of session hijacks; passive, active, and hybrid. [17] The active attack is when the attacker hijacks a session on the network. The attacker will silence one of the machines, usually the client computer, and take over the clients' position in the communication exchange between the workstation and the server. The active attack also allows the attacker to issue commands on the network making it possible to create new user accounts on the network,

which can later be used to gain access to the network without having to perform the session hijack attack.

Passive session hijack attacks are similar to the active attack, but rather than removing the user from the communication session, the attacker monitors the traffic between the workstation and server. The primary motivation for the passive attack is, it provides the attacker with the ability to monitor network traffic and potentially discover valuable data or passwords.

The hybrid attack is combination of the active and passive attacks, which allow the attacker to listen to network traffic until something of interest is found. The attacker can then modify the attack by removing the workstation computer from the session, and assuming their identity. [8]

Most networks attacks depend on software or hardware vulnerabilities as a gateway to an attack.[17] Having knowledge of specific vulnerabilities in these technologies allow the attacker to scan servers to determine what vulnerabilities exist. However, the session hijack attack does not depend on specific software or hardware vulnerabilities, but rather a design limitation within the TCP/IP protocol that does not guarantee security after the connection is made. Session hijacking is also very easy to do, especially on older operating systems [8]. Utilizing commercially available software packages, even a novice computer user has a good chance at successfully waging a session hijack attack. [8].

Stages for Session Hijacking: to clearly understand the state of act of session hijacking we need to identify the vulnerable protocols and also obtain an understanding of what sessions are and how they are used.

Based on research conducted by Mark Lin [11], there are three main protocols that manage the data flow on which session hijacking occurs; TCP, UDP, and HTTP, though other protocols that do not use encryption (e.g. telnet, FTP, DNS) also can be vulnerable.

TCP is an abbreviation for Transmission Control Protocol.

Webopedia [18] defines it as one of the main protocols in TCP/IP networks. Whereas the IP protocol deals only with packets, TCP enables two hosts to establish a connection and exchange stream of data. TCP guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent.

In order to guarantee that packets are delivered in the right order, TCP uses acknowledgement (ACK) packets and sequence numbers to create a full duplex reliable stream connection between two end points, with the end points referring to the communicating hosts. [8].

The connection between client and server begins with a three-way handshake.

After the handshake, it's just a matter of sending packets and incrementing the sequence number to verify that the packets are getting sent and received. The period where all this data is being sent over TCP between client and server is called the TCP session. It is the first stage on which session hijacking will play out.

The next protocol is UDP, which is an abbreviation for User Datagram Protocol.

Webopedia [18] defines it as "a connectionless protocol that, like TCP, runs on top of IP networks. Unlike TCP/IP, UDP/IP provides very few error recovery services, offering instead a direct way to send and receive datagrams over IP networks." [11]

UDP doesn't use sequence numbers like TCP. It is mainly used for broadcasting messages across the network for doing DNS queries. Since it's connectionless and does not have any of the more complex mechanisms that TCP has, it is even more vulnerable to session hijacking. The period where the data is being sent over UDP between client and server is called the UDP session. UDP is the second stage for session hijacking.

HTTP stands for Hyper Text Transfer Protocol.

Webopedia [18] defines HTTP as "the underlying protocol used by the World Wide Web. HTTP defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands. For example, when you enter a URL in your browser, this actually sends an HTTP command to the Web server directing it to fetch and transmit the requested Web page."[11]

It is also important to note that HTTP is a stateless protocol. Each transaction in this protocol is executed independently with no knowledge of past transactions. The result is that HTTP has no way of distinguishing one user from the next. To uniquely track a user of a web application and to persist his/her data within the HTTP session, the web application defines its own session to hold this data. HTTP is the final stage on which session hijacking occurs, but unlike TCP and UDP, the session to hijack has more to do with the web application's implementation instead of the protocol (HTTP).

The Levels

Session hijacking occurs on two levels: [11] the network level and application level. Of the three stages we've defined already, we can classify the TCP and UDP stages under the network level and the HTTP stage under the application level. The levels represent the two different types of

sessions that can be hijacked. The network level refers to the interception and tampering of packets transmitted between client and server during a TCP or UDP session. The application level refers to obtaining session IDs to gain control of the HTTP user session as defined by the web application. Attacks at each level are not unrelated, however, most of the time, they will occur together depending on the system that is attacked. For example, a successful attack on a **Network Level:** network level session hijacking is particularly attractive to hackers, because they do not have to customize their attacks on a per web application basis. It is an attack on the data flow of the protocol, which is shared by all web applications.

Application Level: in the application level, the session hijacker not only tries to hijack existing sessions, but also tries to create new sessions using stolen data. Session hijacking at the application level mainly involves obtaining a valid session ID by some means in order to gain control of an existing session or to create a new unauthorized session.

Session IDs generally can be found in three locations [11]:

- Embedded in the URL, which is received by the application through HTTP GET requests when the client clicks on links embedded with a page.
- Within the fields of a form and submitted to the application. Typically the session ID information would be embedded within the form as a hidden field and submitted with the HTTP POST command.
- Through the use of cookies.

All three of these locations are within the reach of the session hijacker. Embedded sessions information in the URL is accessible by looking through the browser history or proxy server or firewall logs. A hijacker can sometimes re-enter in the URL from the browser history and get

access to a web application if it was poorly coded. Session information in the form submitted through the POST command is harder to access, but since it is still sent over the network, it can still be accessed if the data is intercepted. Cookies are accessible on the client's local machine. The session hijacker has a number of ways to guess the session ID or steal it from one of these locations.

2.1.8 IP Spoofing

IP spoofing is a technique used to gain unauthorized access to computers, whereby the intruder sends messages to a computer with an IP address indicating that the message is coming from a trusted host."[19] The trusted host, in the case of session hijacking, is the client. In employing this technique, the session hijacker obtains the IP address of the client and modifies packet headers to indicate that they come from that IP address. This technique allows the hijacker to create his/her own acceptable packets to inject into the TCP Session. The packets are source-routed, meaning that the sender specifies the route the packet will take to get to the destination IP. Using these source-routed packets, the hijacker can route the packets to his host and fool the server into thinking it is communicating with the victim.

Once the hijacker has successfully spoofed IP address, he determines the next sequence number that the server expects and uses it to inject the forged packet into the TCP session before the client can respond. By doing so, he creates the "desynchronized state".[11] The sequence and ACK numbers are no longer synchronized between client and server, because the server register having received a new packet that the client never sent. Sending more of these packets will create an even greater discrepancy between the two hosts.

2.1.8 Man-in-the-Middle (MITM) Attack

MITM attack is used by malicious internal users to sniff the network traffic between target hosts, in switched networks. [20] Malicious users can tap in on the network traffic for information without the knowledge of the networks legitimate owner. This information can include passwords, e-mail messages, encryption keys, sequence numbers or other proprietary data. Often, some of this information can be used to penetrate further into the network, or cause other severe damage. [19] MITM can also be known as Bucket-brigade attack, Fire brigade attack, Monkey-in-the-middle attack, Session hijacking or TCP hijacking.

Man-in-the-Middle Using Packet Sniffer

When a Man in the Middle (MiM) is performed, a malicious user inserts his computer between the communications path of two target computers.[20] The attack consists of re-routing the network traffic between two target hosts to a malicious host. Then the malicious host will then direct the received packets to the original host, so that the communication between the two target hosts will not be interrupted and the two hosts will not notice that their communication is being sniffed [19]. With all the data between the hosts flowing through the hijacker's sniffer, he is free to modify the content of the packets. The trick to this technique is to get the packets to be routed through the hijacker's host. Lam, LeBlanc, and Smith [16] list two "tricks" that hijackers use to redirect traffic to create this "man in the middle" situation.

The first technique is to use forged ICMP (Internet Control Message Protocol) packets to redirect traffic between client and server through the hijacker's host. [20] ICMP is an extension of IP that is used primarily to send error messages indicating problems procession packets through a connection. In this case, the hijacker is forging messages to fool the client and server into

thinking that the route through his host is better than the original path (better as in faster, shorter, or non-error prone).

The second technique is ARP (Address Resolution Protocol) spoofing. [20] ARP tables are used by each host to map local IP addresses to hardware addresses or MAC addresses. ARP spoofing involves sending out forged ARP replies to fool the host broadcasting the ARP request into updating his ARP table, mapping the IP to be impersonated to the hijacker's hardware address. [4]All traffic intended for that IP will be delivered to his host instead. He can then choose to alter and forward the packets to the real host.

AirJack: is a suite of tools that is designed as a proof-of-concept to establish layer 1 man- in-the-middle attacks against 802.11 networks [10]. Included in this toolset is "wlan-jack", a tool to perform a denial-of-service attack against users on a target wireless network; it works by sending spoofed deauthenticate frames to a broadcast address, purportedly from the network access point's MAC address. Using the wlan-jack tool is simple; an attacker only has to be in range of the clients on the wireless network to be effective. This tool works because the clients that receive the deauthenticate frames believe they have been sent from the access point they are currently associated with. [10]

2.1.9 ARP Cache Poisoning Attack

To map a particular IP address to a given hardware address (MAC address) so that packets can be transmitted across a local networks, systems use Address Resolution Protocol [21]. ARP cache poisoning attack is a method of exploiting the interaction of IP and Ethernet protocols.[19]. This is a malicious act by a host in a LAN. The host introduces a "spurious" IP address to MAC address mapping in another host's ARP cache. ARP operates by sending out

"ARP request" packets. An ARP request asks the question "Is your IP address x.x.x.x? If so, send your MAC back to me." These packets are broadcast to all computers on the LAN, even on a switch network. Each computer examines the ARP request, checks if it is currently assigned the specified IP, and sends an ARP reply containing its MAC address. To minimize the number of ARP requests being broadcast, OS keep a cache of ARP replies. When a computer receives an ARP reply, it will update its ARP cache with the new IP/MAC association. As ARP is a stateless protocol, most OS will update their cache if a reply is received, regardless of whether they have sent out an actual request.[22]

ARP spoofing involves constructing forged ARP replies. By sending forged ARP replies, a target computer could be convinced to send frames destined for computer A to instead go to computer B.[20] The process of updating a target computer's ARP cache with a forged entry is referred to as "poisoning". This can be done by manipulating directly the ARP cache of a target host, independently of the ARP messages sent by the target host. This can be done by either adding a new fake entry in the ARP cache of the target host or by updating an existing entry with a fake IP address to MAC address mapping.

ARP Cache

Each host in a network segment has a table, called ARP cache, which maps IP addresses to their corresponding MAC addresses. There are two entries in an ARP cache; static entries which remains in the ARP cache until the system reboots, mostly used in small LANs and dynamic entries which remains in the ARP cache for few minutes (depending on operating system (OS)) then they are removed if not referenced. This mechanism is used by large networks.

Static ARP cache update

An efficient technique to protect an ARP cache against ARP cache poisoning attack is to use static entries in the ARP cache [20]. The entries in ARP cache cannot be updated by ARP request and reply packets and do not expire. However, in some OS such as windows 2000 and SunOS Solaris 5.9, this is deceitful. Those OS marks static entries in their ARP cache's but authorize their updates by ARP request and reply packets.

Dynamic ARP cache update

To corrupt the entries in the ARP cache of a target host, a malicious host generates an ARP request or reply messages including fake IP and MAC addresses.[20] The success of this depends on the OS of the target host. A malicious host may attempt to send fake ARP reply message to a target host even though the malicious host did not receive ARP request message from the target host. If the OS of the target host accepts the fake ARP reply from the malicious host without checking whether it has send an ARP request message to the malicious host, the fake ARP reply corrupts the ARP cache of the target host with a fake IP/MAC entry. The malicious host can also send an ARP request message instead of an ARP reply message.

An experiment conducted to ascertain which commonly used OS with dynamic entries in the ARP caches are vulnerable to ARP cache poisoning attack indicate that all tested OS accept windows 2000 and Free BSD 4.11, do not allow the creation of new entry by ARP reply messages and all tested OS allows the creation of new entry by ARP request message. Tested OS are windows XP, windows 2000, windows 2003 server, Linux 2.4, Linux 2.6, Free BSD 4.11 and SunOS Solaris 5.9. However, if the entry already exists in the ARP cache, all tested OS allow its update by ARP reply message (even in the absence of ARP request message) or request message

[20]. Therefore using ARP reply message for ARP cache poisoning is difficult to realize against most OS. However it remains possible when using ARP request messages. Malicious users can first use ARP request messages to create fake IP/MAC entries in the ARP caches of their target host. Then, fake ARP fake reply messages are used to maintain existence of the fake entries in the ARP caches of the target host.

2.2.0 The Hacker

The term hacker is somewhat fluid: it is often used by the press to refer to someone who seeks to penetrate a computer system to steal or corrupt data, whereas people who call themselves hackers would reject that definition and use the term to describe someone who is enthusiastic and knowledgeable about computer systems.[1] To avoid this confusion we use the term 'white hat' and 'black hat' (from the days of black and white cowboy films). Thus a 'white hat' hacker might be employed to test a system for flaws, whilst a 'black hat' hacker is synonymous with a cracker. A script kiddie is someone who uses already established and part of automated techniques in attacking a system. Their expertise is less than a hacker, but still considerably more than a normal computer use. A hacker is a person who is able to access other people's computers and modify programs or information. Hacking wireless networks has provided a wider spectrum of victims throughout modern cities. [23].

2.2.1 The Victim

The victims are the sessions that get hijacked. [11]

For TCP and UDP, the sessions are the periods of time where the clients are connected and actively passing information to the server. At the beginning of the session, the user/client is

authenticated and then it is assumed that as long as the ACK numbers on the packets are correct, the server is talking to the same user.

For HTTP, the sessions are the periods of time where the user is accessing a web application, from user logon to user logoff. The HTTP sessions are related to and distinguished from TCP sessions in that "requests" from a single user can come over different TCP/IP streams, directly or through proxies, or even from different IP address." [11] The sessions are all the interaction the user has with the application, regardless of the TCP/IP streams its data travels on. HTTP is a stateless protocol. The result is that web application sessions have to be kept track of separately from the protocol. How this is implemented is dependent on the web application. In general, "when a user logs into an application, a session is created on the server to maintain the state for other requests originating from the same user." [11] The sessions store all necessary parameters and identity information for the particular user it's associated with. It is kept in memory and cached until the user logs out of the application of the application or is inactive for a predefined period of time.

2.3.0 Defenses against Wireless Networks Attacks

Wireless network Security has been under extensive scrutiny in the last few years and has received due attention from both academic and industrial sectors [4].

2.3.1 Network Security Management Plan

An adequate security system management policy has long been an important issue.[22]. A comprehensive network security plan must also consider losses of privacy when we define authentication and authorization as well as losses of performance when we define key management and security protocols. Therefore, a security plan must encompass all of the

elements that make up the wireless and/or wired network, and provide important services such as:

1. Access control, i.e., authorization by capability list, wrappers, and firewalls (access control matrix)

2. Confidentiality, i.e., we must ensure that information and transmitted messages are accessible only for reading by authorized parties.

3. Authentication, i.e., the receiver must be able to confirm that the message is indeed from the right sender

4. No repudiation; the sender cannot deny that the message was indeed sent by him/her.

5. Integrity, i.e., the message has not been modified in transit

6. Availability, i.e., making sure that the system is available to authorized parties when needed

7. Security administration, i.e., checking audit trails, encryption and password management, maintenance of security equipment and services, and informing users of their responsibilities.

2.3.2 Security Measures

Fortunately, there are several ways to protect your computer from the hackers on internet:[23]

Software Updates: software companies are always updating their products, trying to eliminate any security breach that could be exploited by hackers. For this reason, their software makes a periodic check on the latest version available in the internet. It doesn't matter if they are operating systems, office suites, drivers, games or any other kind of specialized software, you always need to assure yourself that every software package is up to date.

Firewall: firewalls became main stream with the Internet. Now-a-days it's impossible to be connected to the net without being checked by hackers, looking for possible breaches in your

connection. Although they can be hardware or software, the most common one among users are the software firewalls, which are installed in your operating system and continually check the transmission of information from your PC to the external world and vice versa.

Antivirus: from all the security packages that a user needs to have installed in his PC, antivirus software is the first. It has been with us for a long time, even before the internet and the now common news on internet hacking. In the old days, virus spread through corporate networks to employee home computers who innocently took work home via diskettes. Finally, it spread to their friends' computers. Today, the Internet has provided viruses a better way to spread themselves through the world. A real time antivirus software package is a must for any user who wishes to navigate safely through the net and not find himself being a victim of a hacker attack.

Data Encryption: all digital mobile systems provide security through some kind of encryption. Data can be encrypted in many ways, but algorithms used for secure data transfer fall into two categories: symmetric and asymmetric. Both rely on performing mathematical operations using a secret number known as a key. The difficulty with symmetric algorithms is that both parties need to have a copy of the key. On the other hand, asymmetric techniques use two separate keys for encryption and decryption. Usually, the encryption key can be publicly distributed, whereas the decryption key is held securely by the recipient.

The most widely used symmetric algorithm in DES (data encryption standard), developed by IBM in 1977. It uses a 56-bit key, which seemed unbreakable at that time. In

1997, a group of Internet users managed to read a DES-coded message. Most organization now use triple-DES, which uses 112 bits. The basic idea is that larger keys mean more possible permutations, and so better encryption.

Several different asymmetric algorithms have been developed, each using a different type of "one-way" mathematical function. Rivest et al. [24] proposed an efficient algorithm, which they refer to as RSA, that relies on the fact that factorization is more difficult than multiplication. Indeed, multiplying two prime numbers together is easy for a computer, but recovering those two numbers from the product is not. The main drawback of asymmetric schemes is that they use a lot of CPU, and so cannot be used to encrypt an entire message through a mobile phone. Instead, it encrypts the message itself using a symmetric algorithm, with a key randomly generated by the network and sent to the handset using an asymmetric algorithm.

2.3.3 IEEE 802.11i Standard

While intended to provide security, the Wired Equivalent Privacy (WEP) [25] protocol lacks good key management and suffers from significant cryptographic problems [26] t o the extent that FBI agents have publicly demonstrated that they can break a 128-bit WEP key in about three minutes. [27] For these reasons, the *IEEE Task Group i* to develop the 802.11i Standard [28], to provide confidentiality, integrity, and mutual authentication. IEEE 802.11i, an IEEE standard ratified June 24, 2004, is designed to provide enhanced security in the Medium Access Control (MAC) layer for 802.11 networks. The IEEE 802.11i wireless networking protocol provides mutual authentication between a network access point and user devices prior to user connectivity. The 802.11i specification defines two classes of security algorithms: Robust Security Network Association (RSNA), and Pre-RSNA. Pre-RSNA security consists of Wired Equivalent Privacy (WEP) and 802.11 entity authentication. RSNA provides two data confidentiality protocols, called the Temporal Key Integrity Protocol (TKIP) and the Counter-mode/CBC-MAC Protocol (CCMP), and the RSNA establishment procedure, including 802.1X

authentication and key management protocols. The protocol consists of several parts, including an 802.1X authentication phase using TLS over EAP,the 4-Way Handshake to establish a fresh session key, and an optional Group Key Handshake for group communications.

Data Confidentiality and Integrity

IEEE 802.11i defines three data confidentiality protocols: Wired Equivalent Privacy (WEP), Temporal Key Integrity Protocol (TKIP), and Counter-mode/CBCMAC Protocol (CCMP).

Authentication and Key Management

Previous work has shown that 802.11 entity authentication (Open System Authentication and Shared Key Authentication) are completely insecure [29], [30]. Therefore, 802.11i defines the Robust Security Network Association (RSNA) establishment procedure to provide strong mutual authentications and generate fresh TKs for the data confidentiality protocols.

RSNA Establishment Procedure

802.11i RSNA establishment procedure consists of 802.1X authentication and key management protocols. Three entities are involved, called the Supplicant (the wireless station), the Authenticator (the Access Point), and the Authentication Server (de facto a RADIUS server [31]). Generally, a successful authentication means that the supplicant and the authenticator verify each other's identity and generate some shared secret for subsequent key derivations. Based on this shared secret, the key management protocols compute and distribute usable keys for data communication sessions. The authentication server can be implemented either in a single device with the authenticator, or through a separate server, assuming the link between the authentication server and the authenticator is physically secure. [31]

2.3.4 Defenses against ARP spoofing

There is no universal defense against ARP spoofing. [19] The only possible defense is the use of static ARP entries. To prevent spoofing, ARP tables would have to have a static entry for each machine on the network. The overhead in deploying these tables, as well as keeping them up to date, is not practical for most LANs. Also of note is the behavior of static routes under Windows. Tests have shown that Windows still accepts spoofed ARP replies and updates the static entry with the forged MAC, sabotaging the purpose of static routes.[19] Aside this, the only remaining defense is detection.

2.4.0 Passive Techniques for Detecting Session Hijacking Attacks in IEEE 802.11 Wireless Networks

Wireless intrusion detection is a challenging research area that is considerably different to and much less understood than, intrusion detection in wired networks.[22] The first challenge facing Wireless Intrusion Detection Systems (WIDSs) is the broadcast nature of the physical (PHY) layer, which makes passive access to the medium a trivial undertaking. Secondly, the limited bandwidth available to wireless physical layers imposes significant efficiency restrictions on intrusion detection techniques. Finally, a wireless network typically consists of mobile client stations like laptops and handheld computers which have limited battery life and computing resources, introducing further constraints on the techniques that may be adopted by a WIDS.

The additional constraints imposed by the wireless environment require that the WIDS not only meet the desirable characteristics of wired IDS, such as a low rate of false positives, but that it must also meet the computational and bandwidth constraints of the wireless environment through

the adoption of efficient detection techniques. Unlike wired intrusion detection systems, a WIDS must also operate to detect and defend against intrusions at the lower PHY and media access control (MAC) layers of the protocol stack. Effective response to intrusions at lower layers in the protocol stack requires a real time monitoring capability, combined with a high level of confidence in the intrusion alerts.

2.4.1 Intrusion Detection Systems

Intrusion is most probably one of the key issues that wireless and mobile systems will have to deal with. [22] The nature of wireless ad hoc networks makes them very vulnerable to an adversary's malicious attacks. Intrusion can be defined as an act of a person or proxy attempting to break into or misuse your system in violation of an established policy.

Today, IDSs, such as Snort [32], have become a standard part of the security solutions, used to protect computing assets from hostile attacks. IDSs are able to detect many types of attacks, such as denial of service (DoS) attacks and IP spoofing attacks.

2.4.2 Wireless Intrusion Detection Systems (WIDSs)

Despite the wide popularity of WLANs, research into WLAN intrusion detection techniques has been limited. [17] Most work concentrates on overall Wireless Intrusion Detection Systems (WIDS) architecture rather than on the quality of the detection techniques employed. Significantly, there are few techniques available for detecting session hijacking attacks that are reliable and effective [17].

Existing approaches for detecting session hijacking attacks include the monitoring of MAC frame sequence numbers. [10], with dramatic changes in sequence numbers indicative of an intrusion; MAC address authentication against lists of valid users and valid wireless network

cards vendors [33], [34], [35], or some combination of the two. Monitoring of received signal strength measurements has also been proposed [17] to permit the detection of signals that may be emanating from unauthorized locations.

Numerous open source WIDS prototypes have been developed, notably Snort-Wireless and WIDZ. Snort-Wireless is capable of detecting rouge APs and STAs, deauthentication and disassociation flood attacks, session hijacking attacks and war driving detection. Rogue detection is based on a list of authorized MAC addresses maintained in a configuration file and the deauthentication/disassociation combined with a broadcast target MAC address while the session hijacking attack detection is based on monitoring for inconsistencies in MAC frame sequence numbers. WIDZ's functionality and techniques are similar to those of Snort-Wireless.

2.4.3 Monitoring Received Signal Strength (RSS)

This detection method was done by Rupinder Gill et al. [17] Received signal strength (RSS) is a measure of the energy observed by the physical layer at the antenna of a receiver. [17] In IEEE 802.11 networks, the RSS indication (RSSI) value is used when performing medium access control clear channel assessments and in roaming operations. The radio frequency (RF) signal strength can be measured in either an absolute (decibel milliwatts-dBm), or relative (RSSI) manner.

The strength of RF signals undergoes some attenuation during transmission after leaving the sender's radio and this signal strength deterioration is governed by a variety of factors like RF interferences, distance between communicating nodes, obstacles etc. The distance between the two nodes has the biggest impact on signal fading. However RF signal strength does not fade in a linear manner, rather it attenuates roughly inversely as the square of the distance between the two

nodes. Along with distance, the RSS for a particular node, as observed by the receiver, also depends on various other factor like the WLAN equipment used by both the sender and the receiver nodes, the physical obstacles in between and their surrounding environment. The mathematical path loss model for IEEE 802.11 RF waves as used by Wullems et. al.[36] also suggests direct relationship between the received signal strength and distance between the sender and the receiver along with numerous other factors, including: frequency used; antenna gain; and an environmental coefficient.

It is not possible for an adversary to accurately guess RSS for a sender as perceived by a receiver. [36] The adversary will need to be at exactly the same location as the receiver, use exactly the same radio equipment, receive the radio signals with same level of interference, reflections and refractions to know the exact RSS value as perceive by the receiver. Even if the sender is stationary, RSS values tend to slightly fluctuate and hence prove almost impossible to guess. This prohibits the adversary from using radio equipment (like a high gain directional antenna) to spoof the RSS as perceived by the receiver.

From an intrusion detection perspective, this property is valuable as it is unspoofable and computationally inexpensive to measure. As it is calculated at the receiver, it is secure from eavesdropping. By periodically monitoring the RSS values for a particular STA or an AP from a passive monitor we can develop a dynamic profile for the communicating nodes based on their RSS values. Any abrupt or unusual changes can be flagged as suspicious activity indicative of a potential session hijacking attack. The RSS profile is dynamic in the sense that it is rebuilt for every session between two nodes and is constantly updated with new observed RSS values for each node per session. Since APs are generally stationary, any abrupt changes in their RSS dynamic profile can probably be flagged as suspicious activity with a higher confidence level.

However, if the STA is mobile then its corresponding RSS values will change more rapidly as observed by the monitor. The uncertainty of the wireless medium thus can be used in favour of intrusion detection, where the adversary has no means of knowing what RSS value to spoof. [17]

2.4.4 Monitoring Round Trip Times of RTS-CTS Handshake

This detection method was done by Rupinder Gill et al. [17] IEEE 802.11 uses both virtual and physical carrier sensing to monitor the state of the medium. [17] Every unicast frame uses its duration field to update the Network Allocation Vector (NAV) of every node in range that receives the frame (wireless medium is broadcast). A node can only transmit data when its NAV is zero. The NAV value reflects the predicted time (in microseconds) it will take to transmit the frame from the sender to the receiver and the corresponding acknowledgement (ACK) frame to return from the receiver to the sender. However another virtual carrier sensing mechanism is used to mitigate collisions from hidden terminals that are not in direct range of the sender and might start transmitting after incorrectly sensing the medium free. Before starting transmission, the sender requests positive control over the medium by sending a Request to Send (RTS) frame to the receiver. On receipt of the RTS frame, the receiver sends a Clear to Send (CTS) frame as an acknowledgment back to the receiver. The duration field in a RTS frame is large enough for the RTS-CTS handshake. All wireless nodes that receive either the RTS or CTS frame update their NAVs and defer access to the medium.

Virtual Carrier sensing ensures that the transmission of a data frame and receipt of its ACK from the receiver is an atomic event, free from collisions. Morrison [10] proposed using this protocol feature to calculate the distance between the sender and the receiver. If the sender monitors the time taken for completion of a data-ACK round trip ($\text{Time}_{\text{rtt}}^{\text{data-ACK}}$)/2 and knowing the speed of

RF waves in air (Speed_{RF}), the sender can easily calculate the distance between itself and the receiver.

This concept can also be extended to the RTS-CTS handshake scenario. Similar to the data-ACK exchange between two nodes, the RTS-CTS handshake is also protected by virtual carrier sensing. In fact RTS-CTS is used to establish the virtual carrier sensing for making the transmission of data frames possible without collisions. The successful receipt of the CTS frame from the receiver indicates that the receiver successfully received the sender's RTS frame and is ready for receiving data. The sender can monitor the time taken for completion of the RTS-CTS handshake between itself and the receiver i.e. $\text{Time}_{s}^{\text{rtt}}$. This is the total time taken for the RTS frame to travel from the sender to the receiver and the CTS frame to be sent back as an ACK. RTS-CTS handshake is free from collisions with other wireless nodes. Hence the only factors that affect the value of $\text{Time}_{s}^{\text{rtt}}$ between two communicating nodes include the distance between the sender and the receiver, the local environment around the nodes and the nature of radio equipment used by both the sender and the receiver.

From an intrusion detection perspective, rapid and abrupt changes in $\text{Time}_{s}^{\text{rtt}}$ between two nodes can be used as a mechanism to detect session hijacking attacks. [17].

2.5.0 Proposed System

We proposed that the source IP address and the source MAC address of every packet coming to a host on a wireless network should be checked against a list of authorized users. That is to capture packets at the destination host using a packet capture, extract the source IP address and the source MAC address and concatenate them to form a single string. The string so constructed is then checked against a list of authorized users on the network. The list of authorized users

consists of the IP addresses and MAC addresses of all the nodes on the network. Therefore any data packet coming from a source machine to a destination machine with a source IP and a source MAC pair, will be expected to match with any of the IP/MAC pair in the list of authorized users, else that data packet is regarded to have come from a malicious host with the MAC address specified in source the Ethernet header.

The proposed system is a passive technique as its run parallel to the NIC's processor and hence does not interfere with the data traffic between communicating nodes. Detection methods like snort are active detection mechanism. This poses an overhead to the NIC's processor which intends delays data traffic between two communication nodes on a wireless network.

Also the passive detection techniques discussed above are not deterministic. For example the detection technique by monitoring RSS depends on abrupt changes in signal strength at the destination node. The strength of RF signal undergoes some attenuation during transmission after leaving the sender's radio and this signal strength deterioration is governed by a variety of factors like RF interferences, distance between communicating nodes, obstacles, rainfall etc. One can not tell when it will rain or when there might be a wind to expect a weaker signal from the receiver. This therefore increases the rate of false positives in the detection mechanism.

Chapter 3

3.0 Software Design

The main objective of the research is to design an application for detecting MITM attack in wireless networks. This has been done by studying and understanding the state of the art of executing MITM attack in order to come out with an effective protocol.

3.1 An Illustration of MITM attack

To do this act, the malicious user should first enable his host's IP packet routing to be able to become a router in order to forward the redirected traffic.[20] Using ARP cache poisoning attack, the malicious user can corrupt the ARP caches of the two target hosts to force the two hosts to forward their packets to his host.

IP routing

By default, IP routing is disabled. This requires a change to the Windows system registry. From the Start menu, click Run. Type regedt32.exe or regedit.exe, and then click OK. In a registry editor, navigate to HKEY_LOCAL_MACHINE \SYSTEM\CurrentControlSet\Services\Tcpip \Parameters Select the IPEnableRouter entry. To enable IP routing for all network connections installed and used by this computer, assign a value of 1. To do this in regedit.exe, right-click the entry, and then click "Modify". In regedt32.exe, click on the wanted entry, click on Edit, and then click on the appropriate menu choice. Close the registry editor.

Let A and B be two users on a network and C be a third user which is trying to intercept the communication between A and B. Host C enables his IP packet routing and corrupt the ARP caches of A and B as shown in the following tables.

Table 3.1 ARP cache of host A before attack

IP address	MAC address
IP_host_B	MAC_host_B
IP_host_C	MAC_host_C

Table 3.2 ARP cache of host B before attack

IP address	MAC address
IP_host_A	MAC_host_A
IP_host_C	MAC_host_C

Table 3.1ARP cache of host A after attack

IP address	MAC address
IP_host_B	MAC_host_C
IP_host_C	MAC_host_C

Table 3.2ARP cache of host B after attack

IP address	MAC address
IP_host_A	MAC_host_C
IP_host_C	MAC_host_C

After the attack, host A associates host B's IP's with the MAC of C and host B associates host A's IP with the MAC of C. Consequently all the packet exchanges between A and B first passes through C. Then host C redirect the packets to the legitimate destination.

3.1.1 Detecting the Illustrated Attack

To execute MITM attack, the malicious host will have to enable its IP packet routing and corrupt the ARP caches of its target hosts. Hence any host with its IP packet routing enabled is a suspected host [20]. We however need to prove that hosts with their IP packet routing enabled have also corrupted the ARP cache of other hosts in the network.

3.1.2 Detecting hosts with IP packets routing enabled

This technique consists of two phase with the assumption that there is a host, Test host in the network used to do all the tests in the two phases. [20]

Phase 1: Generation of trap ICMP (Internet Control Message Protocol) echo request packets: The test host send ICMP ping packet to a given target host in the network. Usually, if a host A with IP address IP_A and MAC address MAC_A, wants to ping another host B with IP address IP_B and MAC address MAC_B in a network, then A has to send to B an ICMP echo request packet. Ping packets are used to determine whether or not a host is connected to a network. When host A wants to send a ping packet to itself, it will set the "Destination MAC address" field in the Ethernet header to MAC_A and the "Destination IP address" field in the IP header to IP_A. The test host will attempt to ping itself, using a trap ICMP echo request but would want to send this packet to all the hosts in the network. Therefore the "Destination MAC address" field in the Ethernet header of the trap ICMP ping packet is set to the MAC address of the target host (MAC_target_host). The value of the "IP address" in the IP header is set to the IP address of the test host (IP_host). Below are the values of the main fields of the trap ICMP ping packet.

Ethernet Header **IP Header**

Source MAC address = MAC_test_host Source IP address = IP_test_host

Destination MAC address = MAC_target_host Destination IP address = IP_test_host

Ethernet Type = 0x0800 (IP message)

ICMP Header

Type = 8 (echo request)

Code = 0

The NIC card of each target host will receive a trap ICMP ping packet sent by the test host. Once the MAC address in the "destination MAC address" field of the trap ICMP packet matches with that of the target host, it will accept the packet and send it to the IP layer for processing. Since the IP address in the "destination IP address" field of the trap ICMP packet does not match that of the target host, it will either discard it or forward it to the host whose IP address is specified in the "destination IP address" field. If the target host is set to do packet routing, then it will route it to the test host otherwise, the trap ICMP packet is discarded. When the target host is set to do IP packet routing, it will forward the original packet with the same IP and MAC headers, but with a different Ethernet header as routers do. The destination MAC address will be that of the test host and the source MAC address will be that of the target host.

Phase 2: Detection of the hosts with enabled IP packet routing: All hosts that will send back the trap ICMP packet to the test host have enabled IP packet routing and hence are considered as suspicious hosts.

3.1.3 Detection of ARP Cache Poisoning Attack

This technique consists of corrupting the ARP cache of the suspicious hosts and forcing them to forward packet received from their victim to the test host. By analyzing the traffic generated by a suspicious host, it is possible to ascertain whether it has performed ARP cache poisoning attack against other hosts in the network.

Let A, B, C, D, and E be hosts in a network. Hosts A and B are transmitting data. Host D has enabled it IP packet routing and it has corrupted the ARP cache of A and B in order to sniff the traffic generated by A and B. The ARP cache of C is not corrupted. Host E is the test host. The detection technique above allows us to identify D as a suspicious host since it has enabled its IP

packet routing. The ARP cache of hosts A, B, C, and D before ARP cache poisoning is as follows:

ARP cache of host A is IP_B-MAC_B, IP_C-MAC_C, IP_D-MAC_D, ARP cache of host B is IP_A-MAC_A, IP_C-MAC_C, IP_D-MAC_D, ARP cache of host C is IP_A-MAC_A, IP_B-MAC_B, IP_D-MAC_D and ARP cache of D is IP_A-MAC_A, IP_B-MAC_B, IP_C-MAC_C.

After ARP cache poisoning attack, the ARP caches of the various hosts are as follows:

ARP cache of A (corrupted ARP cache) is IP_B-MAC_D, ARP cache of B (corrupted ARP cache) is IP_A-MAC_D, ARP cache of C is IP_A-MAC_A, IP_B-MAC_B and ARP cache of D is IP_A-MAC_A, IP_B-MAC_B.

For each suspected host, we corrupt its ARP cache using ARP cache poisoning attack. To do that, the test host E, send a fake ARP request to the suspicious host D, so that all the entries of the ARP cache of host D will have the MAC address of the test host E as MAC_E.

Consequently, all packets send by A to B will first go the suspected host, D since all the ARP caches of A and B have been corrupted by D. Suspicious host D will then redirect the traffic to the test host, E since the ARP cache of D has been corrupted by the test host E. The test host will send a copy of the received packet to its original destination, B as illustrated in fig 3.1 in the next page.

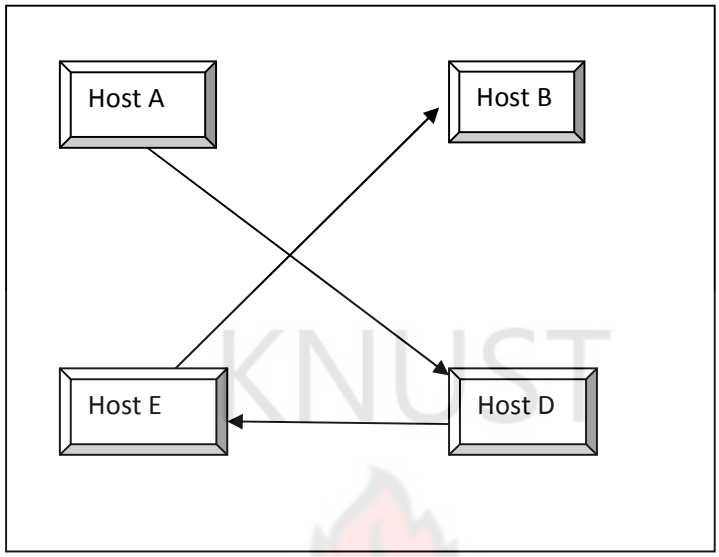


Figure 3.0 Detection of ARP Cache Poisoning Attack

By analyzing the packet sent by the malicious host, D to the test host, E we can deduce that the source IP address in the IP header of the packet is of host A, but the source MAC address in the Ethernet header of the packet is of the malicious host D, when it should have been that of host A. The next section presents a detail design of the software.

3.2.0 Preliminary Study

The rationale behind the design is to have a mechanism for checking the source MAC address and the source IP address against a list of authorized users on a WLAN. The goal of this rational is to flag out any unusual IP address/MAC address mapping and hence detecting the host with that particular MAC address as a malicious host.

There is no such thing as perfect security; we need to consider how a system will react to a successful attack. The main constrain of the study is that the system can only detect man-in-the-middle attack but cannot prevent malicious host from waging this attack on other users on the

network. Detecting such a malicious activity and reporting the detection to the network administer for necessary action is however worthwhile. The benefit to the users on the network is that the system will contribute towards the maintenance of secure network systems in wireless networks.

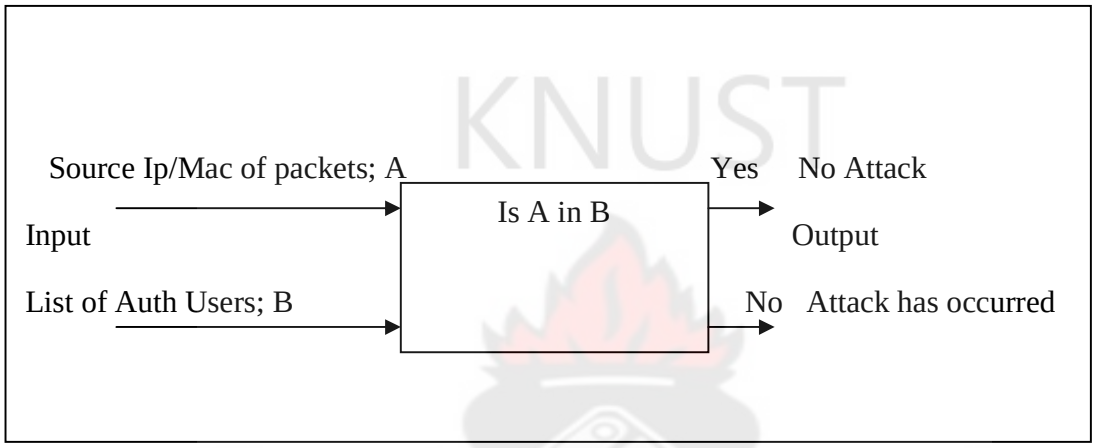


Figure 3.1 Basic system components

3.2.1Analysis

Users of this system were legitimate hosts on a wireless network. The system will extract the source IP address and the source MAC address of every packet coming to destination host. It will concatenate these strings into a single string and the string so constructed will be checked in a list of database containing the source IP addresses and the source MAC addresses of authorized users on a wireless network. If this string exits will be an indication of “No Man-in-the-Middle Attack” occurred else “Man-in-the-Middle Attack” has occurred. The system will be used in a wireless network environment and it will be operational when the various hosts are sharing data.

3.2.2 Detail Software Design

The software was mainly procedural with one function. The software designed is required to open a list of captured packets. That is, packets captured online using a Network Protocol

Analyzer (Wireshark) was printed into a text file and subsequently normalized into another file. The normalized text file contains the source IP addresses and the source MAC addresses of all packets coming to the destination host.

The opened normalized text file was read into and the source IP address and the source MAC address were extracted for each packet in the list.

The software is required to concatenate these two strings into a single string. This constructed string is the search string.

The software also opened a list of authorized users on the WLAN. This list was generated by the network administrator which consists of the IP address/MAC address pairs of all the authorized users on the network.

The software is required to open an empty text file where the results of the simulation were printed into.

The string so constructed (search string) was then searched against the list of authorized users opened. If this string exists in the list, “No Man-in-the-Middle Attack” was printed into the results text file; else “Man-in-the-Middle Attack from the host with that MAC address” was printed into the results text file.

3.2.3 Logic of the Design

By analyzing the packets emanating from other host to the destination host, we can deduce weather the packet came from the right source or not. The logic is to create a unique string of every host on the network. Therefore concatenating the IP address and the MAC address of legitimate hosts to form a single string by the network administrator, there is no such combination that can be formed by a malicious host on a network in order to carry out malicious activities. The reason is that, for a host on a network to forward traffic to another host, the logical address of the receiving host should be specified in the ARP cache of the sending machine. Also

for a malicious host to fool another host on a network to direct traffic to the malicious host intended for another host, then the logical address of the rightful destination should be specified in the ARP cache of the sending machine whist the physical address of the malicious host should be specified in the ARP cache of the sending machine. This superiors mapping will not however be in the database of authorized users.

Therefore any IP/MAC pair that will not be in the list of authorized users will be considered as a malicious user on the network.

3.2.4 Design Architecture

First, the list of captured packets is opened. Then the source MAC address and the source IP address are extracted and concatenated to form a string called Source_mac_ip. Source_mac_ip so constructed is checked against a list of Source_mac_ip of authorized users on the network. If the constructed Source_mac_ip is found in the list, “Success” is printed else “Alert: Man-in-the-Middle” would be printed. The flowchart on the next page indicates the design procedure.

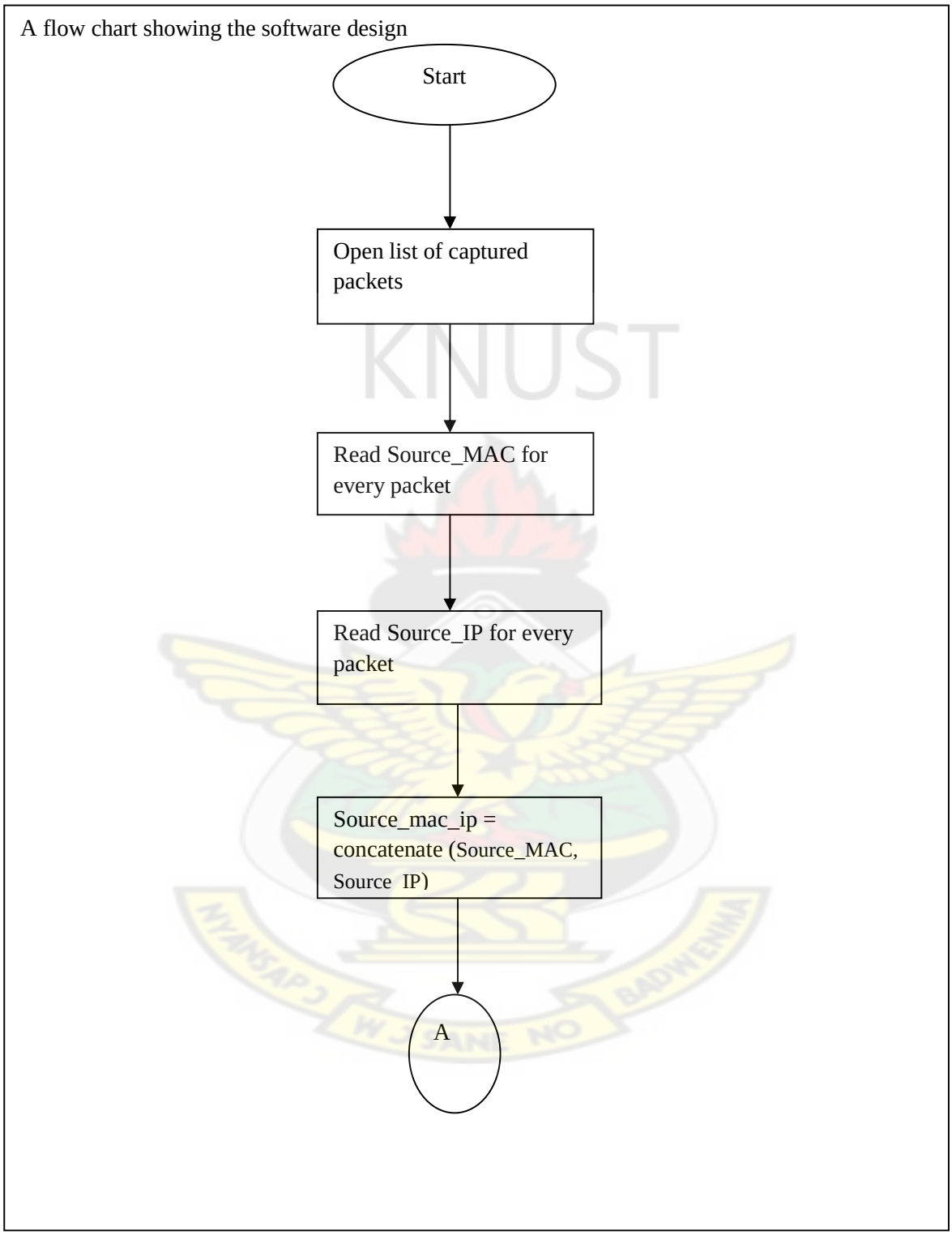
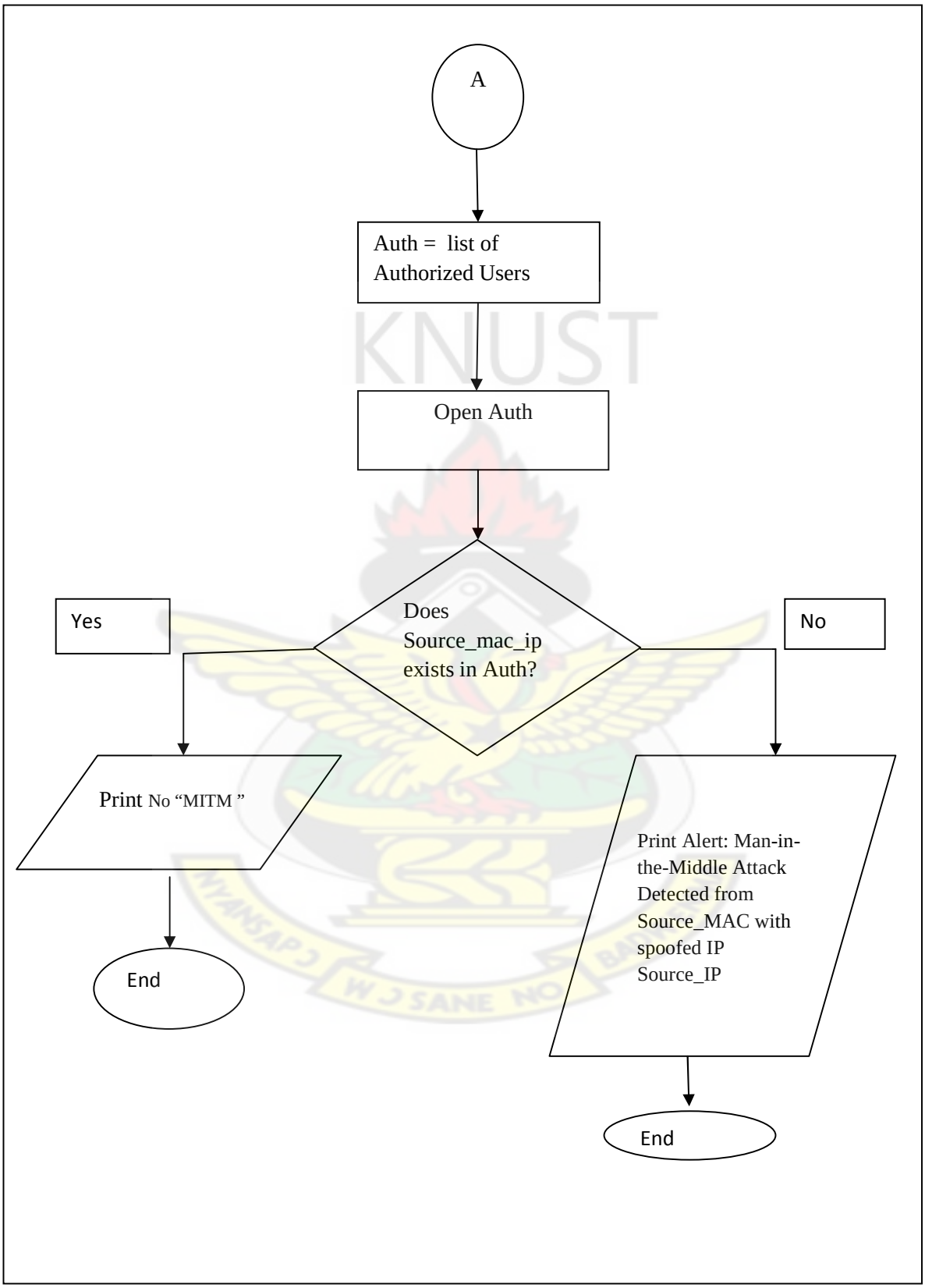


Figure 3.2 shows a flow chart explaining the logic of the software design


A
Auth = list of Authorized Users
Open Auth
Does Source_mac_ip exists in Auth?
Yes
No
Print No "MITM "
End
Print Alert: Man-in-the-Middle Attack Detected from Source_MAC with spoofed IP Source_IP
End

Chapter Four

Implementation and Testing

4.0 Implementation

From the state of the art of executing man in the middle attack as shown in figure 3.1 and detection method discussed above, it is realized that by analyzing every packet arriving at the receiving host, we can detect a malicious hosts on IEEE 802.11 networks.

4.1Tools used

i) ActivePerl 5.101

ii) Network Protocol Analyzer (Wireshark)

iii) At least three laptops

iv) Colasoft Packet Builder 1.0

4.2 Experimental setup

Three laptops with wireless enabled were used to create MITM scenario. Computer A and B were the nodes to initiate the traffic and Computer C will be the 'man in the middle'. Colasoft Packet Builder 1.0 was used to create and send packets from computer A. Computer B was the node to receive the traffic. Computer A sent packets with an IP, IP_A, for some time. Computer C then sent packets with an IP, IP_A (instead of IP_C) whiles computer A is taken out of the network. A network protocol analyzer, Wireshark was used to capture data packets arriving at Computer B. The captured packets were analyzed.

The interface on the next page indicates the packet capture.

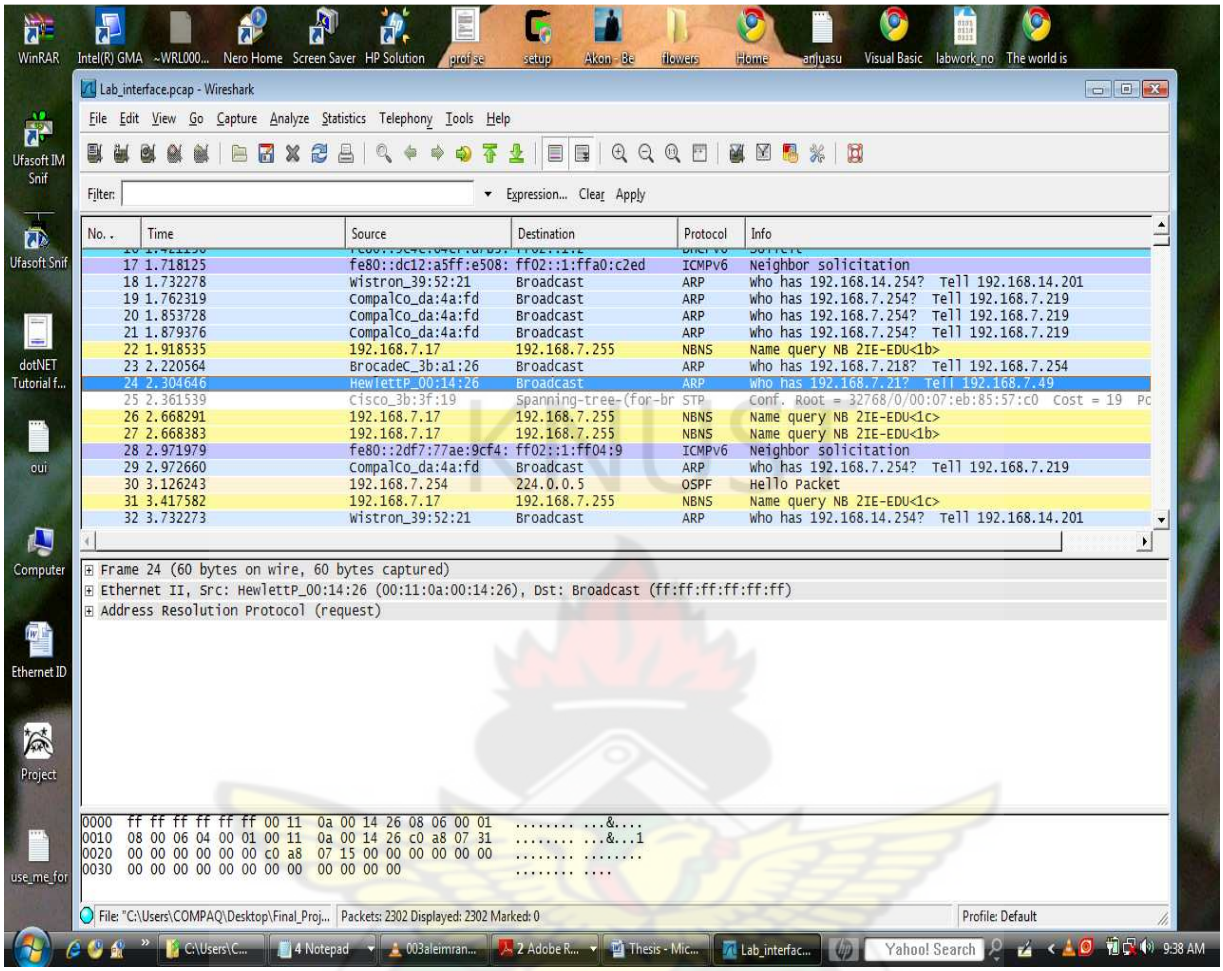


Figure 4.0 captured packets.

Wireshark is a network packet analyzer. A network packet analyzer captures network packets and then displays details about the contents.

Looking from left to right on the top half of the display you will first see a number assigned to each packet starting with 1. The next column displays the exact amount of time from the start of Wireshark until the packet was captured. The next two columns display the source and destination IP address in IPv4 or IPv6 format, depending on which format is being used by the network operating system. In the fifth column the protocol is identified such as ARP,

DNS, DNS, DHCP, etc. The last column on the right will provide a brief bit of information about each packet captured. Some examples of the information will be things such as a request to

logon, who is at a specific IP address, DHCP information, response to another packets and much more. For detailed information about a packet, you simply select the packet with the mouse and the bottom half of the display presents you with detailed information and a list of protocols used to encapsulate each packet. In the sample in Figure 4.0 you can see that frame 24 is an Ethernet type II packet encapsulating an Address Resolution Protocol (ARP) request with source IP address, Hewlettp_00:14:26 (00:11:0a:00:14:26). The packet is a broadcast packet with destination address, ff:ff:ff:ff:ff:ff. This is a very common protocol encountered on Ethernet networks.

From this, we realized that the mac address, 00:26:55:bb:25:09 was mapping to two different IP addresses, 192.168.7.17 and 169.254.2.2.

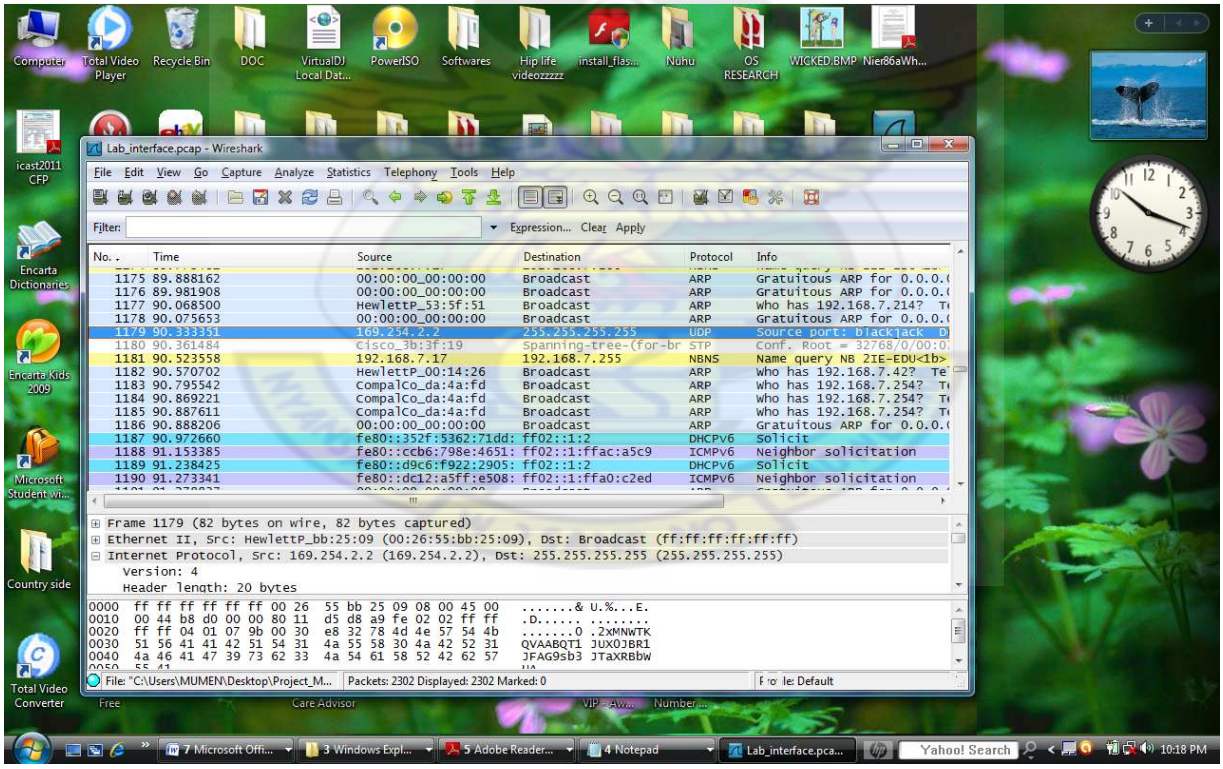


Fig 4.1 Capture interface showing an anomaly on frame number 1179

From figure 4.1, frame number 1179 has source IP address, 169.254.2.2 mapping to a MAC address 00:26:55:bb:25:09. Also from figure 4.2 below, frame number 1181 has a source IP address, 192.168.7.17 mapping to a MAC address 00:26:55:bb:25:09.

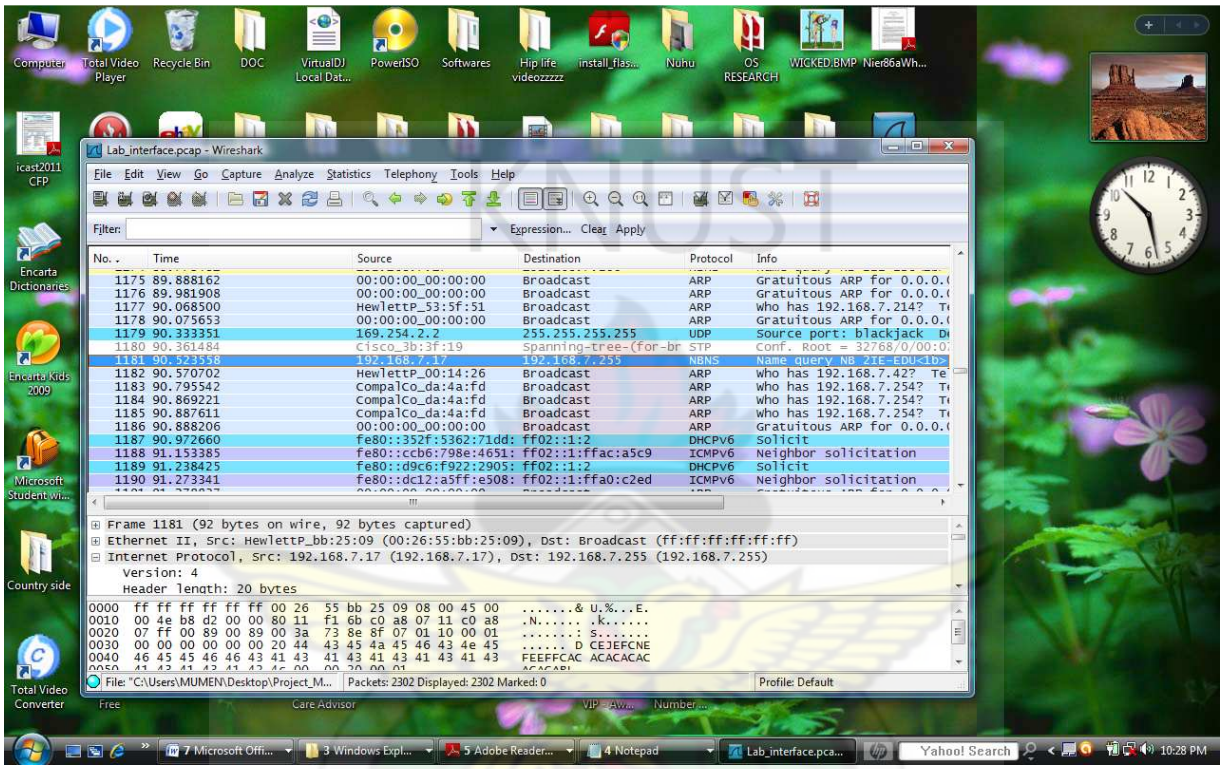


Figure 4.2 Capture interface showing an anomaly on frame number 1181

The captured packets were printed into a text file called labwork_hijacking which was then normalized and read into to retrieve the source mac address and the source ip address. The figure on the next page presents the printed captured packets.

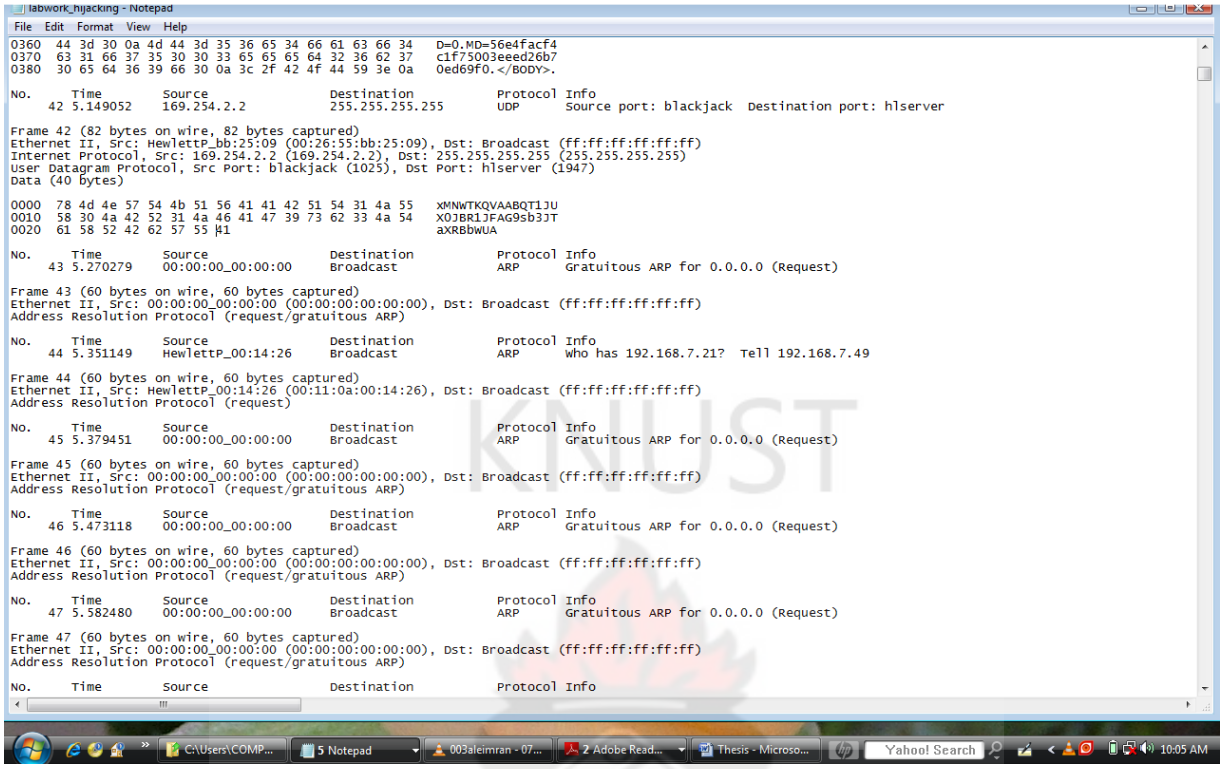


Fig 4.3 Printed captured packets

The source MAC address and the source IP address were extracted from figure 4.3 into a text file called lab_work.txt. The figure on the next page shows the data. The software was developed to read into this file for the source MAC and source IP.

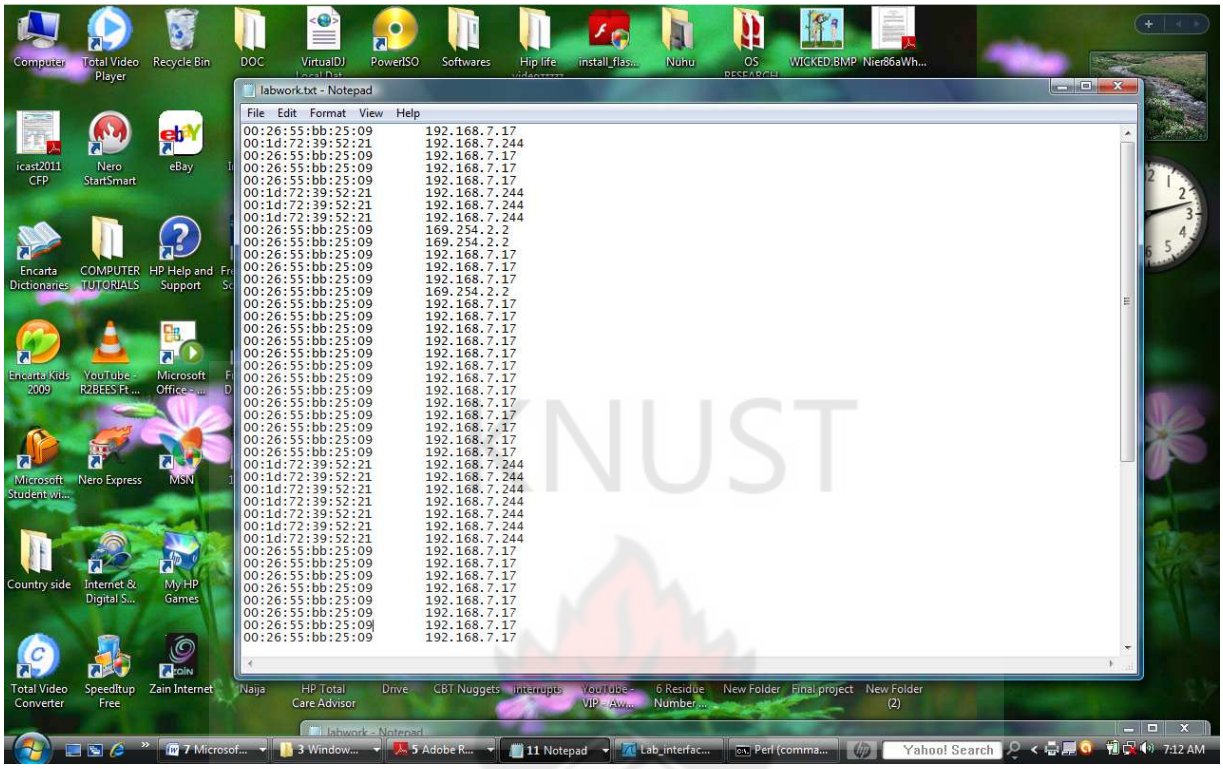


Figure 4.4 Normalized text showing source MAC and source IP

As shown in figure 4.4, each line begins with the source MAC address which is in six octets. The next four doted decimal represent the source IP address. The source MAC address and IP address were concatenated to form string which was then checked against a list of authorized users on the network. The figure on the next page shows a list of authorized users that was used for the simulation process.

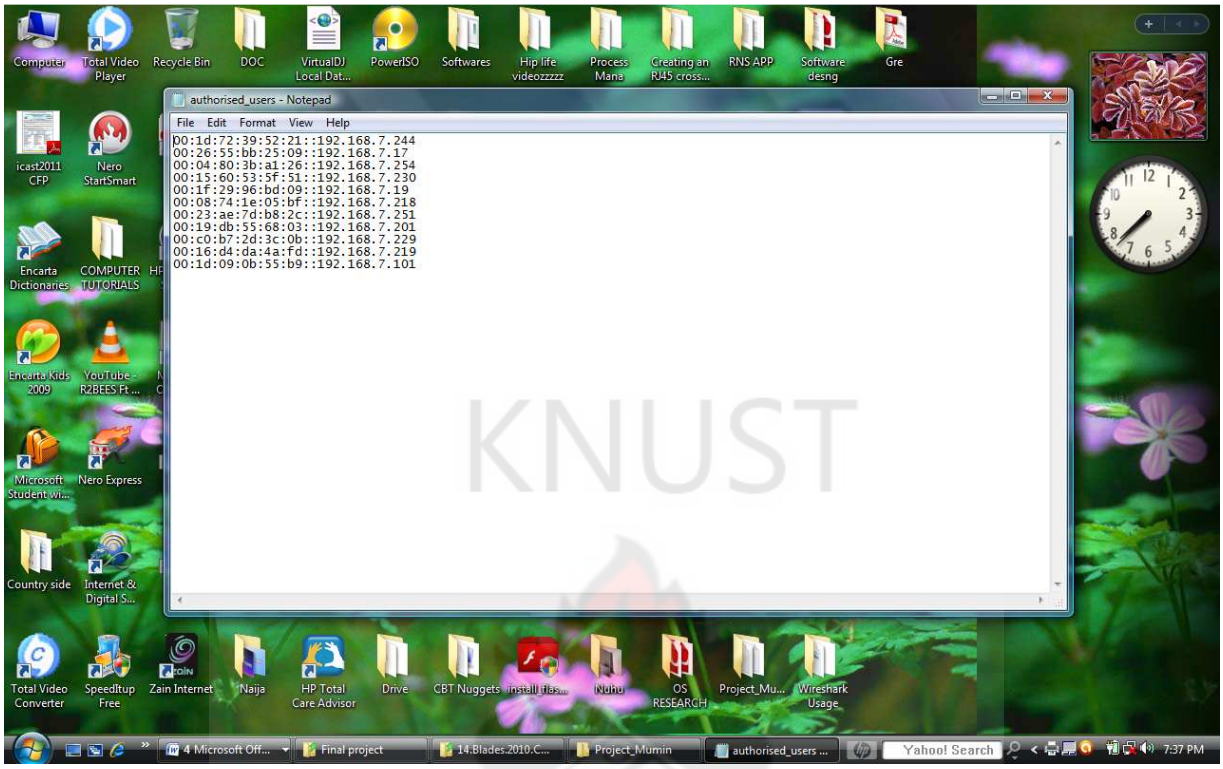


Figure 4.5 A list of authorized users.

As shown in figure 4.5, this network consists of 11 legitimate users. The MAC address of every host is paired with its legitimate IP address. So any MAC/IP pair which was not found in this list was regarded as a malicious host.

4.3 Testing

The software was developed using ActivePerl 5.101. Perl supports object-oriented, procedural and functional programming. From its introduction to the programming community in 1987, Perl has become today one of the most widely known and used programming languages. Designed by Larry Wall, and originally thought of as a natural enhancement for the popular *csh* shell script notation of Unix, Perl was at first primarily used for text manipulation. Its maturity in the early 90's coincided with the rise of the Web, and it rapidly became the most popular programming

language for HTML form processing and other Web development as well. Perl is (an) interpreted language optimized for scanning arbitrary text files, extracting information from those text files, and printing reports based on that information. It's also a good language for many system management tasks. The language is intended to be practical (easy to use, efficient, complete) rather than beautiful (tiny, elegant, minimal). It combines some of the best features of C, sed, awk, and sh. Expression syntax corresponds quite closely to C expression syntax.

The software was run on varying network sizes. For example, it was tested on network of three hosts and also on a network of 11 hosts.

The figure below shows the output of the simulation

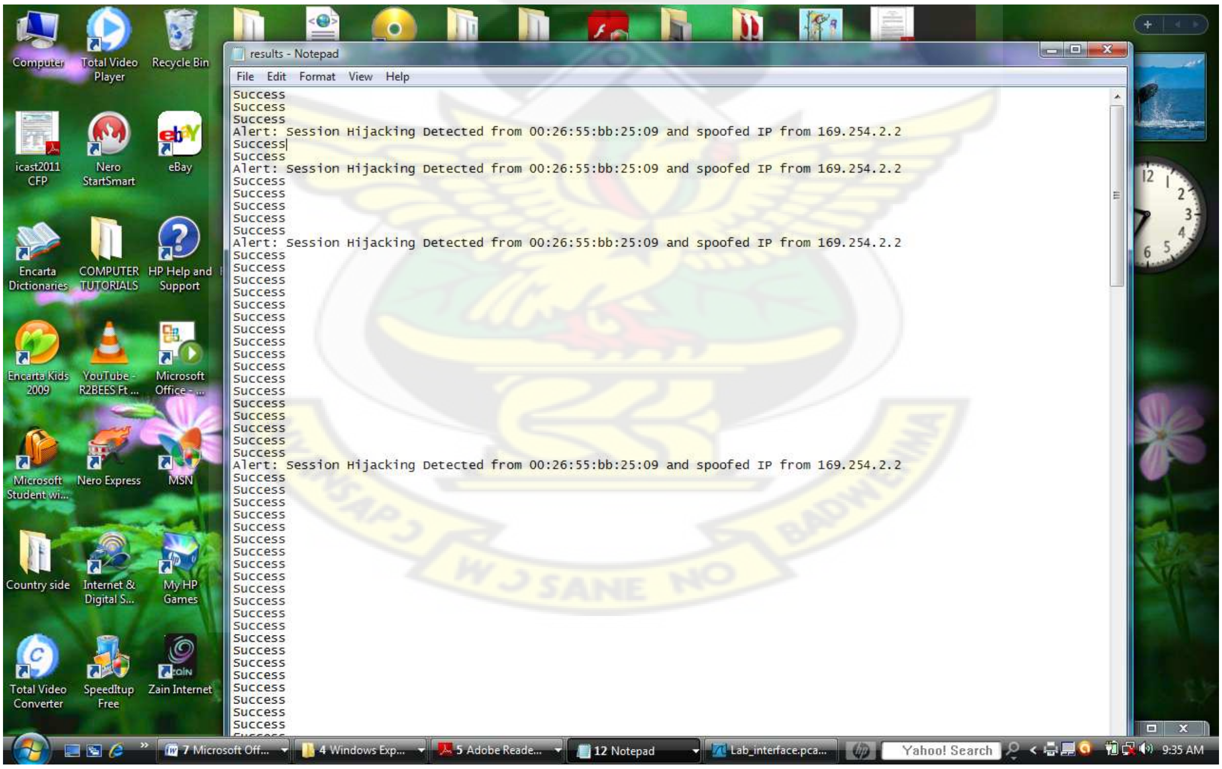


Figure 4.6 Results of simulation

4.4 Evaluation

The performance of the proposed detection mechanism is evaluated against monitoring of Received Signal Strength (RSS). RSS is a measure of the energy observed by the physical layer at the antenna of a receiver. From intrusion detection perspective as suggested by Rupinder Gill, et al, by periodically monitoring the RSS values for a particular STA or an AP from a passive monitor we can develop a dynamic profile for the communicating nodes based on their RSS values. Any abrupt or unusual changes can be flagged as suspicious activity indicative of a potential session hijacking attack.

However the strength of RF signals undergoes some attenuation during transmission after leaving the sender's radio and this signal strength deterioration is governed by a variety of factors like RF interferences, rainfall, wind, intervening obstacles etc. These factors can cause the signal strength to change abruptly without necessary passing through an adversary as some of these factors are natural occurring phenomenon. This detection mechanism can not indicate who is actually stealing the session.

Also Rupinder Gill, et al also argue that the distance between sender and adversary must be the same as the distance between the sender and receiver before the adversary can deduce the signal strength the receiver will be expecting, so he can re-route the stolen data packets to the receiver without the receiver noticing that the data packets actually came the adversary. However the adversary can form an equilateral triangle with the sender and the receiver such that the distance between the sender and the adversary will be the same as the distance between sender and the receiver.

However the proposed detection mechanism is not affected by these factors as it was proven to be deterministic based on the simulations results above. Also, apart from being a detection

mechanism, it was able to detect which of the nodes on the network actually carried out the malicious activity.

Chapter Five

Conclusion and Recommendations

5.1 Conclusion

Based on the testing done in chapter four which was conducted in varying network sizes, the detection mechanism was able to identify any hosts with a MAC address with an IP address that is supposed to be for a different host in the network.

Also from the evaluation performed against an existing detection mechanism, monitoring of RSS, we have been able to show that our proposed detection method is a better option. It is therefore concluded that the proposed detection mechanism was effective in detecting man-in-the-middle attack in IEEE 802.11 networks.

5.2 Recommendations

The tools used for this project were used in isolation, which did not give a very practical illustration of the proposed detection mechanism. Future works should therefore consider integrating all these tools for a more practical realization of the proposed detection mechanism. Having been able to integrate all these tools will enable organizations like financial institutions, academic institutions, military organizations etc to use this detection method to maintain a more secured system for their wireless networks.

It is also recommended that a different software be developed that will automatically generate the list of authorized users within the network. This will eliminate the manual system of generating that list by the network administrator, hence making it user friendly. This will further make the proposed detection mechanism applicable in a network with a dynamic host

configuration protocol server enabled, where a network administrator will not be needed to maintain the detection system

It is further recommended that online packets should be captured and online methods to the analysis made on the captured packets. This will make the detection system more robust and real time.

References

1. Andrew Simmonds, Peter Sandilands, Louis van Ekert: An Ontology for Network Security Attacks
2. Bhavin Bharat Bhansali -Passive Techniques for Detecting Session Hijacking Attacks in IEEE 802.11Wireless Networks
3. Javvin Technologies: Network Security Dictionary
4. Wikipedia -Man-the-middle attack
5. Whatis.com| SearchSecurity.com Definitions: What is man in the middle attack?
6. Melani| Information Assurance Reporting and Analysis Center: Semi-Annual Report 2005 Issue 2
7. Akshaya Bhatia -Man-the-Middle Attack
8. Paul Jess advised by Richard Winner -Session Hijacking in Windows Networks, 12th October 2006
9. Mary Landesman About.com Guide- Man in the Middle attack and Redirection
10. J. D. Morrison. IEEE 802.11 wireless local area network security through location Joshua Wright -Detecting Wireless LAN MAC Address Spoofing, GCIH, CCNA, 21/01/2003
11. Mark Lin -An overview Session Hijacking at Network and Application Level Submitted 1/1812005 GSEC Practical Assignment v1.46c (option 1)
12. Changhua He John C Mitchell: Security Analysis and Improvements for IEEE 802.11i
13. Changhua He John C Mitchell: Security Analysis and Improvements for IEEE 802.11i
14. IEEE P802.11i/D10.0. Medium Access Control (MAC) security enhancements, amendment 6 to IEEE Standard for local and metropolitan area networks part 11:

Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications. April 2004

15. Rajani Muraleedharan and Lisa Ann Osadciw: Jamming Attack Detection and Countermeasures In Wireless SensorNetwork Using Ant System
16. Lam, K. & Leblanc, D. & Smith, K., (2006). Theft on The Web: Prevent Session Hijacking. Retrieved from the web 07/06/2009
17. Rupinder Gill, Jason Smith, Mark Looi and Andrew Clark. Information Security Institute, Queensland University of Technology, Brisbane, Australia
18. Webopedia- Internet protocols
19. Sean Whalen -Introduction to ARP Spoofing , April, 2001
20. Zouheir Trabelsi and Khaed Shuaib -Spoofed ARP Packets Detection in Switched LAN Networks
21. Plummer, D.C.: An Ethernet Address Resolution Protocol-Converting Network Protocol to a 48 bit Ethernet Address for Transmission on Ethernet Hardware, RFC-826 (1982)
22. Azzedine Boukerche -Security and Fraud Detection in Mobile and Wireless Networks
23. nospysoftware.com, Protect against all kinds of spyware
24. R. Rivest, The MDS message-digest algorithm, RFC286, Internet Engineering Task Force,Symbolic, Inc., 1982
25. IEEE Standard 802.11-1999. Local and metropolitan area networks - specific requirements - part 11: Wireless LAN Medium Access Control and Physical Layer specifications. 1999

26. N. Borisov,I . Goldberg,an d D. Wagner. Intercepting mobile communications: the insecurity of 802.11. In Proceedings of the 7th Annual International Conference on Mobile Computing and Networking
27. H. Cheung. FBI Teaches Lesson in how to break into Wi-Fi networks. Information Week Network Pipeline, 2005
28. IEEE P802.11i/D10.0. Medium Access Control (MAC) security enhancements, amendment 6 to IEEE Standard for local and metropolitan area networks part 11: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications. April 2004
29. W. A. Arbaugh, N. Shankar, and J. Wang. Your 802.11 Network has no Clothes. In Proceedings of the First IEEE International Conference on Wireless LANs and Home Networks, pages 131-144, December, 2001.],
30. N. Borisov,I . Goldberg,an d D. Wagner. Intercepting mobile communications: the insecurity of 802.11. In Proceedings of the 7th Annual International Conference on Mobile Computing and Networking,
31. C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial In User Service (RADIUS). *RFC 2865*, June, 2000
32. http://www.snort.org
33. M.K. Chirumamilla and B. Ramamurthy. Agent based intrusion detection and response system for wireless LANs. In2003
34. W.-C. Hsieh, C.-C. Lo, J.-C. Lee, and L.-T. Huang. The implementation of a proactive wireless intrusion detection system.2004.
35. H. Yang, L. Xie, and J. Sun. Intrusion detection for wireless local area network. , 2004

36. C. Wullems, K. Tham, and M. Looi. Proximity-based network packet filtering for IEEE 802.11 wireless devices.2004

Appendix

```
#!/usr/bin/perl -w
use strict;
use Getopt::Long;
use IO::Handle;
my (
$progname,
$line, $i, $k,
$source_mac, $source_ip, $source_mac_ip,
$athorise_use,
);
my (
%athorise_use, %source_mac_ip2,
);
my (
@sou_mac_iparr, @array,
);
$progname = $0;
$progname =~ s,.*/,,; # only basename left in progname
$progname =~ s/\.\w*$//; # strip extension if any
# Main
#Open the list of packets captured
print "Reading list of packets captured..\n";
open ( FILE, "C:\\Users\\COMPAQ\\Desktop\\Final_Project\\labwork.txt" ) or
   die "Cannot open file: $!";
```

```
print "Done.\n";
my ($line2);
while ($line2 = <FILE>) {
$source_mac = substr($line2, 0, 17);
#chomp $source_mac;
 $source_ip = substr($line2, 17, 14);
#chomp $source_ip;
$source_mac =~ s/\s//g;
$source_ip =~ s/\s//g;
print "$source_mac\n";
print "$source_ip\n";
@sou_mac_iparr = ($source_mac, $source_ip);
$source_mac_ip = join('::', @sou_mac_iparr); #40
print "$source_mac_ip\n";
print "Reading list of authorised users..\n";
open ( FILE1, "C:\\Users\\COMPAQ\\Desktop\\Final_Project\\authorised_users.txt" ) or
   die "Cannot open file: $!";
print "Done.\n";
open ( FILE2, ">C:\\Users\\COMPAQ\\Desktop\\Final_Project\\good_results.txt" ) or
   die "Cannot open file: $!";
print "Done.\n";
my (@line);
while (@line = <FILE1>) {
```

```
my $look_for = $source_mac_ip;
my @array = @line;
 $i = 0;
while (($i <= $#array && ($array[$i] ne $look_for))) {
  ++$i;
}
if ($i <= $#array) { # searched it
  print FILE2 "No Session Hijacking dectected! \n";
} else {
print FILE2 "Success\n";
}
}
}
close FILE;
close FILE1;
close FILE2;
```