Security Analysis & Implementation of Session Hijacking on Different Operating Systems

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Mohit
Roll No. 801032015

Under the supervision of:

Dr. Maninder Singh
Associate Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

June 2012

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Security Analysis & Implementation of Session Hijacking on Different Operating Systems*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Dr. Maninder Singh and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Mohit

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr. Maninder Singh)
Associate Professor
Computer Science & Engineering Department
Thapar University
Patiala

Countersigned by

(Dr. Maninder Singh)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

First of all, i am grateful to The Almighty for establishing me to complete this thesis. Many people have shared their time and expertise to help me accomplish my goal. First, I would like to sincerely thank my guide, Dr. Maninder Singh (Associate professor, Computer Science and Engineering Department) for his inspiration, guidance, immense help and support throughout the period of this research work. He has provided me with all the necessary resources and research environment without which it would not have been possible to complete this work. It was a great opportunity for me to do this work under his supervision.

I am deeply indebted to my dearest family for their love and encouraging moral support, which enabled me to pursue my studies.

I would also like to thank Mr. Balvinder Singh (Lab Supdt) for providing me with the material support

I can not skip mentioning the help which my friend Mr. Ravinder Dahiya, has provided throughout my work. I am also very thankful to the entire faculty and staff member of Computer Science and Engineering Department for their direct-indirect help, cooperation, love and affection which made my stay at Thapar University memorable

Mohit

(801032015)

Abstract

Computer networks, both public and private are essential in today's life. Security is the big issue for all networks in todays environment. There is always a struggle between user friendliness and security and the downside to the user friendliness in these kind of networks is poor security. Session Hijacking is still a problem of network security. The aim of session hijacking is to hijack the user account without the acquaintance of user-name or password. This attack can be done in both wired or wireless networks. If the vulnerability exists which leads to session hijacking then session hijacking is very serious problem because normal users can't maintain their privacy. In order to do session hijacking no special software is needed. Wireless network are more vulnerable than wired network. Session hijacking is directly dependent upon the network sniffing. Hub or wireless network is more prone to passive sniffing. Passive sniffing is difficult to detect. Application layer (HTTP) session hijacking attack can be done by taking advantages of vulnerability exists in either networks or Web accounts.

The practical part of this thesis starts with the setup of a test bench with three types of computers, an attacker, legitimate client and a LAMP server. Passive and active sniffing attack was performed on the network. By using LAMP sever this thesis evaluates the result of security analysis of session hijacking on different operating systems. Whether any operating system is vulnerable to session hijacking attack. The main conclusion is that the risk of session hijacking attack is significantly reduced with the help of some free software. This thesis also discuss the basis and limitation of HTTP session hijacking attack.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Network Security

A computer network, often simply referred to as a network, is a collection of computers and devices interconnected by communications channels that facilitate communications and allows sharing of resources and information among interconnected devices[1]. A computer network consists of many different parts, including network device such as firewalls, routers and switches. Computer network has played a magnificent role in bringing the world at the minimum difference of a key press. These networks connect together in many different ways to form the single entity that we know as the Internet. Internet acts as information superhighway. The Internet has undoubtedly become the largest public data network, enabling and facilitating both personal and business communications worldwide.

With the explosion of the public Internet and e-commerce, private computers, and computer networks, if not adequately secured, are increasingly vulnerable to damaging attacks. All computer users, from the most casual Internet surfers to large enterprises, could be affected by network security breaches. The meaning of network security is different for different persons. Network security involves all activities that organizations, enterprises, and institutions undertake to protect the value and ongoing usability of assets and the integrity and continuity of operations. An effective network security strategy requires identifying threats and then choosing the most effective set of tools to combat or mitigate them.

Two basic approaches used to deal with network security vulnerabilities: proactive and reactive.

Proactive Approaches

A proactive system constantly tests the organization's network for vulnerabilities and exposures. All IP devices attached to the network are periodically or continuously scanned. The attacks comes under this category are Zero day attacks [2].

Reactive Approaches

Reactive systems, such as intrusion detection or intrusion prevention systems depend on an attack, incident, A company only reacts if it recognizes that it's been attacked. A organization should have proactive approaches for DDOS attack, phishing attack, E-mail spoofing, Virus attack, Session hijacking attack.

1.2 Popular attacks on a Network

1.2.1 Sniffing

A sniffer is an application or device that can read, monitor, and capture network data exchanges and read network packets. If the packets are not encrypted, a sniffer provides a full view of the data inside the packet. By placing a packet sniffer on a network in promiscuous mode[3], a malicious intruder can capture and analyze all of the network traffic. Within a given network, username and password information is generally transmitted in clear text which means that the information would be viewable by analyzing the packets being transmitted. Even encapsulated (tunneled) packets can be broken open and read unless they are encrypted and the attacker does not have access to the key.

1.2.2 IP spoofing

IP spoofing is a technique used to gain unauthorized access to computers, where by the attacker sends messages to a computer with a forging IP address indicating that the message is coming from a trusted host. IP address that is used within the range of trusted IP addresses.

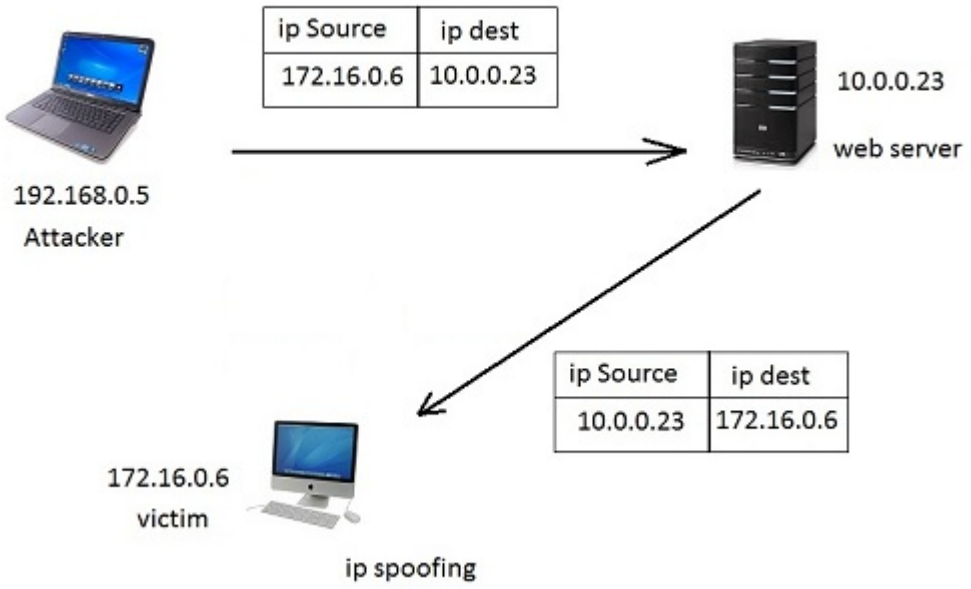


Figure 1.1: IP spoofing attack

1.2.3 Denial-of-Service Attack

A denial-of-service attack (DoS attack) or distributed denial-of-service attack (D-DoS attack) a type of attack on a network that is designed to bring the network to its knees by flooding it with useless traffic[4].

1.2.4 Phishing

Phishing is a way of attempting to acquire information such as usernames, passwords, and credit card details by masquerading as a trustworthy entity in an electronic communication. Most people associate phishing with e-mail messages that spoof, or mimic, banks, credit card companies or other business like Amazon and eBay. These messages look authentic and attempt to get victims to reveal their personal information. But e-mail messages are only one small piece of a phishing scam[5].

1.2.5 Social Engineering

Social Engineering is art of convincing people to reveal confidential information. Social engineering depend upon the fact that people are unaware of their valuable

information and are careless about protecting it[6].

1.2.6 Trojan

A Trojan is a malicious program misguided as some very important application. Trojans comes on the backs of other programs and are installed on a system without the user's knowledge. Trojans are malicious pieces of code used to install hacking software on a target system and aid the hacker in gaining and retaining access to that system. A Trojan may give a hacker remote access to a targeted computer system. Once a Trojan has been installed on a targeted computer system, hackers may be given remote access to the computer allowing them to perform all kinds of operations[7].

1.2.7 Spyware

Spyware is a type of malware (malicious software) installed on computers that collects information about users without their knowledge. The presence of spyware is typically hidden from the user and can be difficult to detect. Spyware is often secretly installed on a user's personal computer without their knowledge. However, some spyware such as keyloggers may be installed by the owner of a shared, corporate, or public computer on purpose in order to intentionally monitor users[8].

1.2.8 Viruses

A computer virus is a type of malware that propagates by inserting a copy of itself into and becoming part of another program. It spreads from one computer to another, leaving infections as it travels. Viruses can range in severity from causing mildly annoying effects to damaging data or software and causing denial-of-service (DoS) conditions. Almost all viruses are attached to an executable file[9].

1.2.9 Worms

Computer worms are similar to viruses in that they replicate functional copies of themselves and can cause the same type of damage. In contrast to viruses, which

require the spreading of an infected host file, worms are standalone software and do not require a host program or human help to propagate. To spread, worms either exploit a vulnerability on the target system or use some kind of social engineering to trick users into executing them. A worm enters a computer through a vulnerability in the system and takes advantage of file-transport or information-transport features on the system, allowing it to travel unaided[10].

1.2.10 Bots

"Bot" is derived from the word "robot" and is an automated process that interacts with other network services. Bots often automate tasks and provide information or services that would otherwise be conducted by a human being. A typical use of bots is to gather information (such as web crawlers), or interact automatically with instant messaging (IM), Internet Relay Chat (IRC), or other web interfaces. They may also be used to interact dynamically with websites. It's hard to determine the precise size of the bot problem. According to Symantec, which measures bot activity as a 14-day moving average, 800,000 to 900,000 PCs at any given time are zombies infected with some type of bot[11].

1.2.11 Buffer Overflow

A buffer overflow occurs when a program or process tries to store more data in a buffer (temporary data storage area) than it was intended to hold. Since buffers are created to contain a finite amount of data. The extra information, which can overflow into adjacent buffers, corrupting or overwriting the valid data held in them. Although it may occur accidentally through programming error, In buffer overflow attacks, the extra data may contain codes design to trigger specific action, in fact sending new instruction to attacker computer that could, for example, damage the user's files change data or disclose secret information[12].

1.2.12 Session Hijacking

It refers to the exploitation of valid computer session where the session between two computers is taken over. The attacker steals the valid session id of other

computer.

Session hijacking can be done at 2 levels:

(a) Network level (TCP and UDP session hijacking)

(b) Application level (HTTP session hijacking)

(a) **Network level session hijacking (TCP session hijacking)**:- When an attacker takes over tcp session between two computers. In this a hacker watch the sequence number and acknowledge number in IP packet transmitted between two computers by using sniffing program.

TCP sequence number prediction :- TCP sequence number prediction is used by attackers to attack TCP sessions, and takes advantage of the fact that TCP is a sequenced data delivery protocol. TCP segments are encapsulated within IP datagrams and as a result there is no guarantee that the datagrams will follow the same route and therefore arrive in the order they were sent.

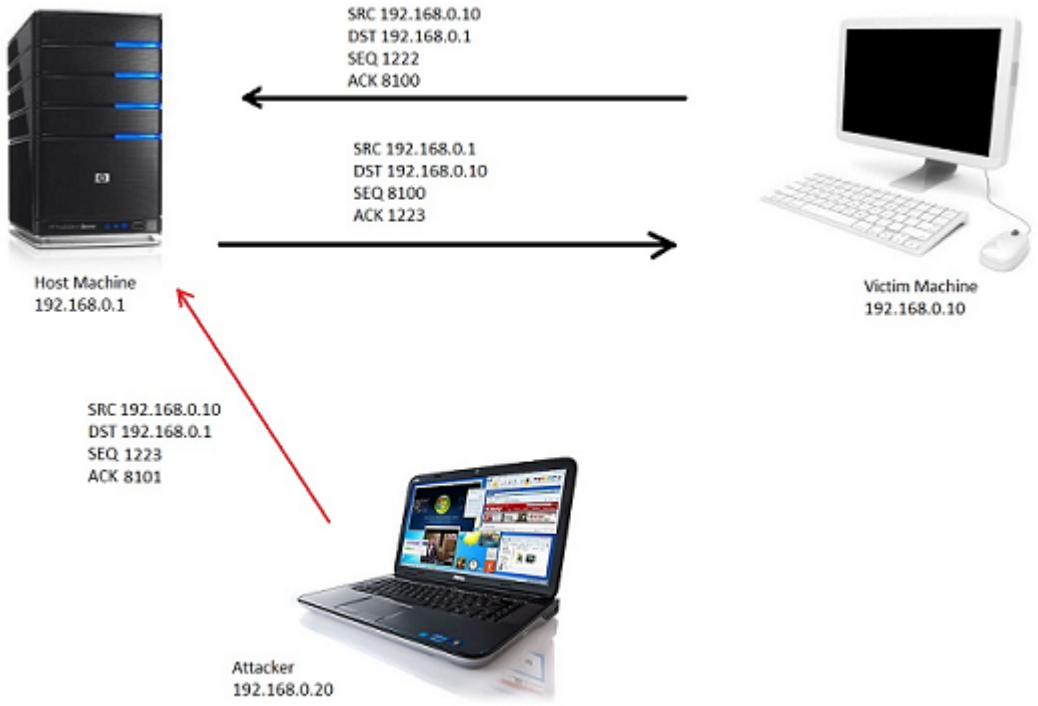


Figure 1.2: Sequence number prediction by attacker

The only requirement is that the attacker has access to the IP datagrams sent between the target and a spoofed host as this is necessary to obtain the correct

sequence number. When a sequence number is predicted, a TCP segment can be sent; effectively taking over the connection and all further packets sent by the spoofed host will be ignored by the target host because the sequence numbers will be incorrect. TCP session hijacking has a number of benefits over other attacks, such as sniffing IP datagrams for passwords, especially when advanced identification and authentication techniques are in use. All advanced authentication techniques happen at connection time, no protection is afforded by them after this point. Therefore, the attacker simply hijacks a legitimate connection to gain entry to a system[13].

(b) **Application level (HTTP session hijacking)**-: Application level is about gaining control on the HTTP's user session by obtaining the session id.

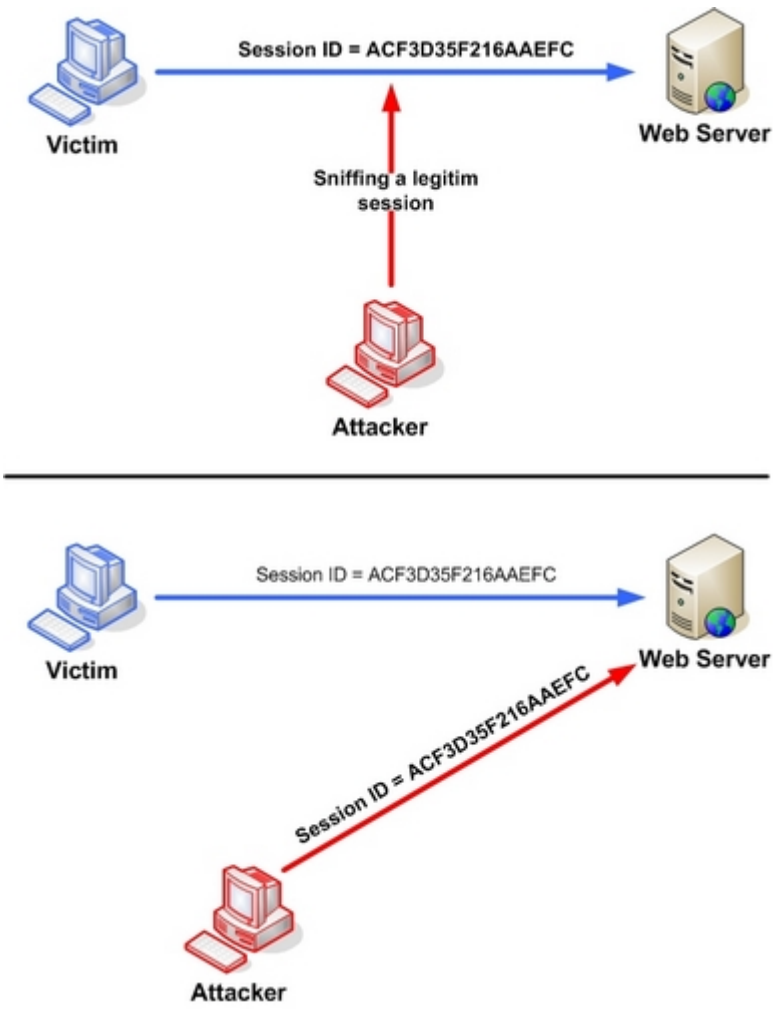


Figure 1.3: Sniffing and stealing session id by attacker

Session id -: session id is unique number that is assign or given by web server to a specific user to identify the session. Every time an website is visited, a new session id is assigned. Closing a browser and then reopening and visiting the site again generates a new session id. Session id can be stored as a cookie, form field, or URL (Uniform Resource Locator). Session id also refers as session identifier or session token[14]. Secure connection and encrypted cookie prevent the user to being victim of session hijacking. But many website use HTTPS (secure socket layer) to prevent their user from eavesdropping. Sometimes all content of that website does not use secure connection or sometimes website use their HTTPS security only for username name and password not for after when session is fixed.

1.2.13 Session Fixation

In a session fixation attack, the attacker fixes the user's session id before the user even logs into the target server, thereby eliminating the need to obtain the user's session id afterwards. First, the attacker who in this case is also a legitimate user of the system logs in to the server[15].

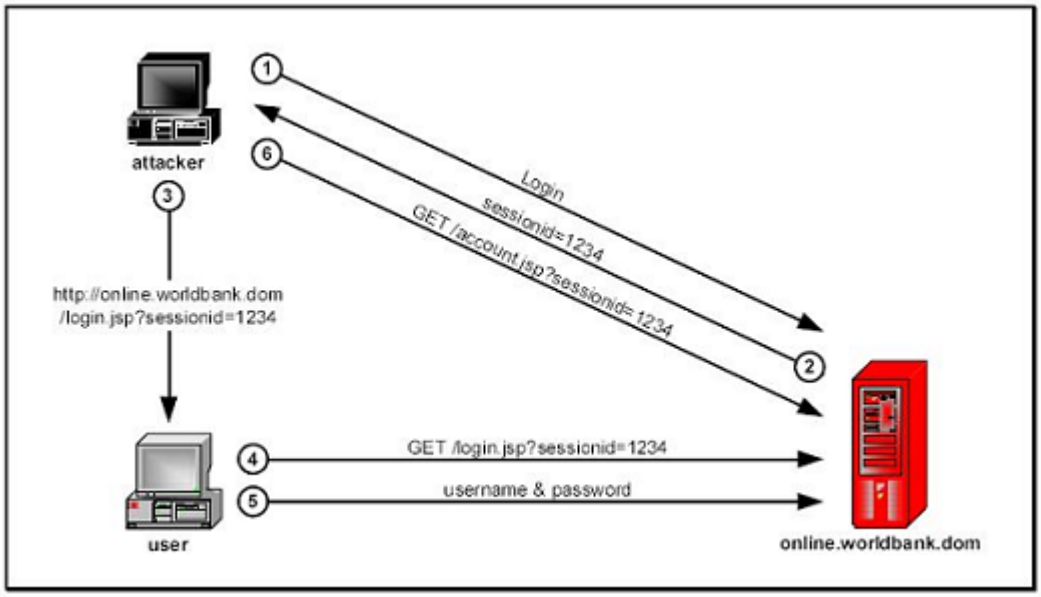


Figure 1.4: Session Fixation [14]

(1) Server is issued a session ID 1234.

(2) Then sends a hyperlink http://online.worldbank.dom/login.jsp?sessionid=1234 to the user, trying to lure him into clicking on it.

(3) The user (how convenient for our example) clicks on the link, which opens the server's login page in his browser.

(4) The user provides his credentials to the login script.

(5) The server grants him access to his bank account. However, at this point, knowing the session ID, the attacker can also access the user's account via account.jsp?sessionid=1234

1.2.14 Cross Site Scripting

Cross Site Scripting (also known as XSS or CSS) is generally believed to be one of the most common application layer hacking techniques. In general, cross-site scripting refers to that hacking technique that leverages vulnerabilities in the code of a web application to allow an attacker to send malicious content from an end-user and collect some type of data from the victim[16]. The security community has already developed numerous proof-of-concept demonstrations in which XSS holes in websites such as Hotmail, eBay and in software like Apache ,Tomcat, Microsoft IIS and IBM Websphere facilitate hijacking of web application user accounts[17].

Today, websites rely heavily on complex web applications to deliver different output or content to a wide variety of users according to set preferences and specific needs. This arms organizations with the ability to provide better value to their customers and prospects. However, dynamic websites suffer from serious vulnerabilities rendering organizations helpless and prone to cross site scripting attacks on their data. Cross Site Scripting allows an attacker to embed malicious JavaScript, VBScript, HTML, or Flash into a vulnerable dynamic page to fool the user, executing the script on his machine in order to gather data. The use of XSS might compromise private information, manipulate or steal cookies, create requests that can be mistaken for those of a valid user, or execute malicious code on the end-user systems. The data is usually formatted as a hyperlink containing malicious content and which is distributed over any possible means on the Internet.

Chapter 2

Literature Survey

Literature survey refers to following sections in the field of session hijacking. In this chapter login sessions, Session management Method, Some of the common WLAN vulnerabilities and a brief overview of the SSL handshake are described.

2.1 login session

In computing, a login session is the period of activity between a user logging in and logging out of a (multi-user) system. Login session is created when valid user-name and password is entered. Session is created by a unique number this unique number is called Session id or session token. Web-browser store session id and this id is checked on each pages visited by user. If session id is valid then user can access the web-pages related to their account. When logout function is called then server kill the session of user. Session creation depends upon the parameters used in programming. Parameter can be user-name, password and current time. It is bad idea to included password as parameter. However, session information is temporary and will be deleted after the user has left the website[18]. Algorithm of session creation should not be weak.

Figure 2.1 shows that session is created after 1st step. HTTP session hijacking is done in between 2nd and 3rd steps. Session id is matched on every page visited by user. Session id is resided in HTTP packet. HTTP packet can be captured by either active sniffing or passive sniffing. In a typical web application login s-cenario, two authentication tokens are exchanged a user-name and password for

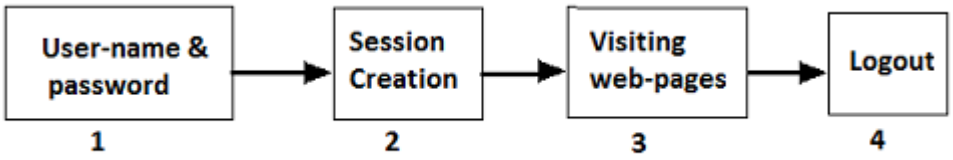


Figure 2.1: Steps of session creation and logout

values stored in a cookie. A cookie is a small piece of information sent by a Web server to store with a Web browser so that it can later be read back from that browser. Cookies may collect a unique identifier, user preferences, and profile and membership information from a user, thereafter used as the only authentication token. It is commonly understood that a user's web session is vulnerable to hijacking if an attacker captures that user's cookies Session id is contained by the cookie. Cookie can be extracted from that HTTP packet. By injecting cookie in web-browser, web-browser can direct open web-pages of victim users without entering the user-name and password[19].

2.2 Session Management Method to Improve Web Applications

In the traditional web applications, the valid duration of session is set short enough to decrease risk of session hijacking and to release memory of unnecessary session objects as soon as possible. When the access time exceeds the valid duration and the session expires, the clients of the web applications must send requests and data all over again from the beginning. The resubmission decrease easy to use web applications. On the mobile network, we encounter such situation frequently. To solve this problem researchers propose a new session management method that manages the valid duration of session and data respectively in separate objects instead of managing them in one object together. researchers compare the performance of proposed method with that of conventional method and show that the proposed method realizes the convenience on the mobile network[20].

HTTP session hijacking attack is very easy to conduct. HTTP session hijacking attack can be performed by taking advantages of the vulnerabilities of networks ,

E-mail accounts and Operating systems.

Session Hijacking is very vulnerable in Wireless network. Wireless network is not as much secure as wired network. Anyone can sniff the traffic on the wireless network.

2.3 Some Common Wireless LAN Vulnerabilities

2.3.1 Accidental association

Unauthorized access to company wireless and wired networks can come from a number of different methods and intents. One of these methods is referred to as accidental association. When a user turns on a computer and it latches on to a wireless access point from a neighbouring company's overlapping network, the user may not even know that this has occurred. However, it is a security breach in that proprietary company information is exposed and now there could exist a link from one company to the other. This is especially true if the laptop is also hooked to a wired network.

2.3.2 Easy to eavesdrop

Because Wireless uses the airwaves, it is easy to listen in on network traffic or even connect to a network. However, just listening to network traffic does not necessarily produce results if the data is encrypted with strong encryption. If WEP encryption is used it is more likely that hackers can, with some effort decrypt the information they have intercepted[21].

2.3.3 Inadequate encryption standards

Wired equivalent privacy (WEP) is a weak encryption standard to say the least, therefore some users will not even enable it. This can prove to be detrimental to the wireless LAN because weak encryption is better than no encryption at all. Also, some users make the encryption key longer but this does not make the

LAN more secure, it just makes the hacker work harder at trying to penetrate the system. WEP's major weakness is its use of static encryption keys[22].

2.3.4 Identity theft (MAC spoofing)

Identity theft (or MAC spoofing) occurs when a cracker is able to listen in on network traffic and identify the MAC address of a computer with network privileges. Most wireless systems allow some kind of MAC filtering to only allow authorized computers with specific MAC IDs to gain access and utilize the network. However, a number of programs exist that have network sniffing capabilities. Combine these programs with other software that allow a computer to pretend it has any MAC address that the cracker desires, and the cracker can easily get around that hurdle[21].

2.3.5 Rogue access points

These may be illicit access points brought in to the enterprise by employees, or poor access point setup by the untrained employee described above. An employee might also mistakenly use SOHO access points that are not designed to be used in an enterprise because of its weak security options. Other rogues may include external malicious users such as hackers engaging in war driving in an attempt to access the wireless LAN from nearby locations[23].

2.3.6 Secure Configuration of Authorized Access Points

Organizations also need to ensure that all authorized wireless access points are securely configured. It is especially important to change all default settings because they are well known and can be exploited by attackers[23].

2.3.7 Passive Scanning

Scanning is the act of sniffing by tuning to various radio channels of the devices. A passive network scanner instructs the wireless card to listen to each channel for a few messages. This does not reveal the presence of the scanner.

An attacker can passively scan without transmitting at all. Several modes of a station permit this. There is a mode called RF monitor mode that allows every frame appearing on a channel to be copied as the radio of the station tunes to various channels. This is analogous to placing a wired Ethernet card in promiscuous mode. A station in monitor mode can capture packets without associating with an AP or ad-hoc network. The so-called promiscuous mode allows the capture of all wireless packets of an associated network. In this mode, packets cannot be read until authentication and association are completed[24]. An example sniffer is Kismet[25].

An experimental study of HTTP session hijacking attack on top 10 Free Web-Mail had been conducted by Preecha Noiumkar Thawatchai Chomsiri Mahasarakham University Maha sarakham, Thailand[26]. This research presents the results of the experimental about security level of the Top 10 popular Free Web- Mail. These 10 Web Mails were hacked by means of Session Hijacking. The researcher conducted this experiment on the LAN system and used information capturing technique to gain Cookies and Session ID inside Cookies. Then, Hijacking was conducted by using two Hijacking methods. The first method, which was common and easy to conduct, used only one Cookie. The second method, which was not very popular but offered high penetrating power, used all Cookies (Cookies cloned by SideJacking tools). The results show that the Web Mails with the height security level are AOL Mail, GMX Mail and Yahoo Mail; and the Web Mails with the low security level are Gmail, Inbox Mail and Hotmail.

2.4 An overview of the SSL handshake

The SSL protocol runs above TCP/IP and below higher-level protocols such as HTTP or IMAP. It uses TCP/IP on behalf of the higher-level protocols, and in the process allows an SSL-enabled server to authenticate itself to an SSL-enabled client, allows the client to authenticate itself to the server, and allows both machines to establish an encrypted connection.

Step 1. The SSL client sends a “client hello” message that lists cryptographic in-

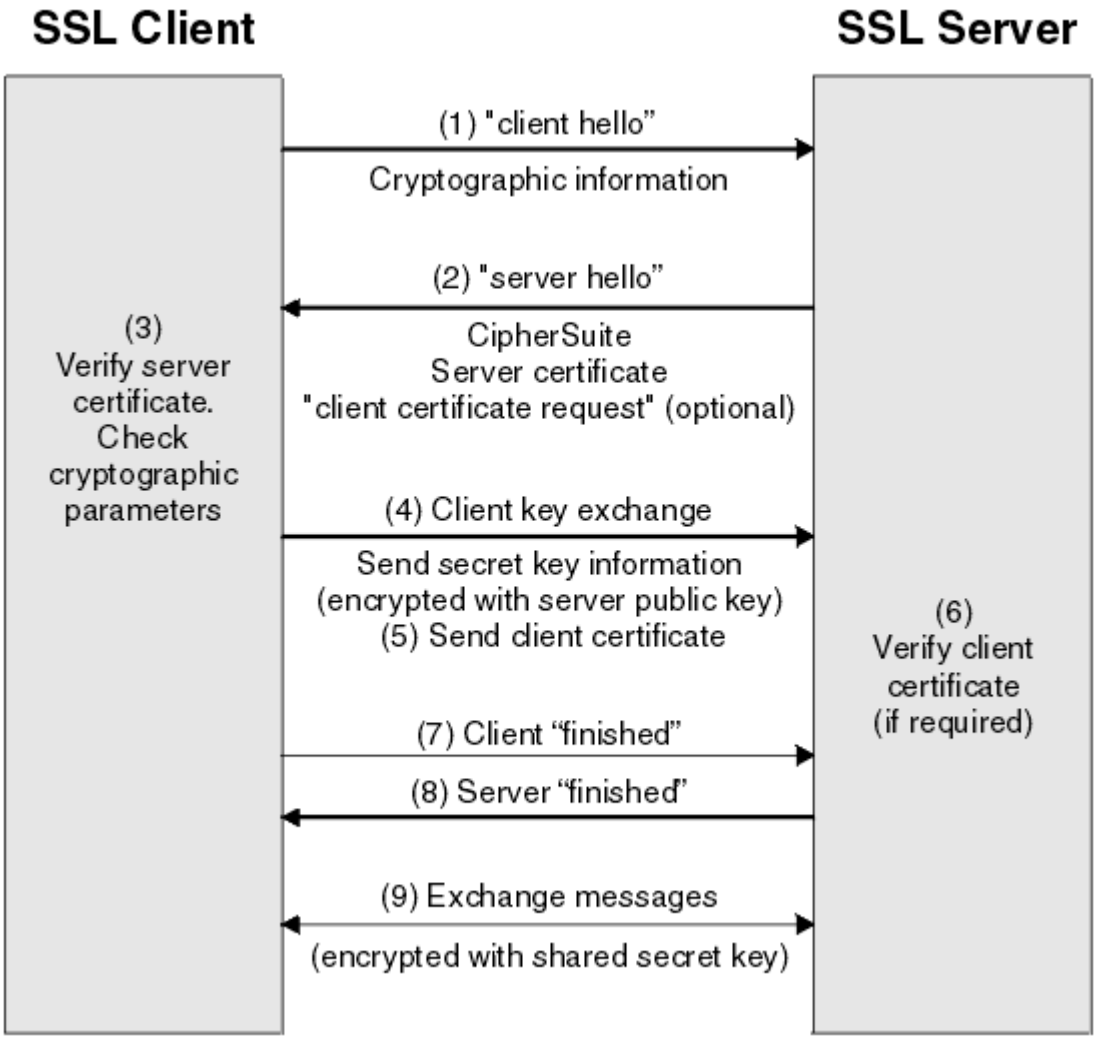


Figure 2.2: SSL handshake

formation such as the SSL version and, in the client's order of preference, the CipherSuites supported by the client. The message also contains a random byte string that is used in subsequent computations. The SSL protocol allows for the "client hello" to include the data compression methods supported by the client, but current SSL implementations do not usually include this provision.

Step 2. The SSL server responds with a "server hello" message that contains the CipherSuite chosen by the server from the list provided by the SSL client, the session ID and another random byte string. The SSL server also sends its digital certificate. If the server requires a digital certificate for client authentication, the server sends a "client certificate request" that includes

a list of the types of certificates supported and the Distinguished Names of acceptable Certification Authorities (CAs).

Step 3. The SSL client verifies the digital signature on the SSL server's digital certificate and checks that the CipherSuite chosen by the server is acceptable.

Step 4. The SSL client sends the random byte string that enables both the client and the server to compute the secret key to be used for encrypting subsequent message data. The random byte string itself is encrypted with the server's public key.

Step 5. If the SSL server sent a "client certificate request", the SSL client sends a random byte string encrypted with the client's private key, together with the client's digital certificate, or a "no digital certificate alert". This alert is only a warning, but with some implementations the handshake fails if client authentication is mandatory.

Step 6. The SSL server verifies the signature on the client certificate.

Step 7. The SSL client sends the SSL server a "finished" message, which is encrypted with the secret key, indicating that the client part of the handshake is complete

Step 8. The SSL server sends the SSL client a "finished" message, which is encrypted with the secret key, indicating that the server part of the handshake is complete.

Step 9. For the duration of the SSL session, the SSL server and SSL client can now exchange messages that are symmetrically encrypted with the shared secret key.

A site must be completely hosted over HTTPS, without having some of its contents loaded over HTTP or the user will be vulnerable to some attacks and surveillance and redirection from HTTP to HTTPS connections would be security risk [27].

In 2003, Peter Burkholder analyzed SSL handshake defect and verify the possibility of attack [28] to SSL.

2.5 HTTPS connection initiated by HTTP

According to analysis of user's habits and the practical applications of HTTPS, HTTPS request will be initiated by the following two ways:

a. **Users's habits:** When user accesses HTTPS sites using web browser, user usually type directly the URL without HTTPS in the address bar of web-browser, such as: www.yyy.com. If there is no protocol specified in URL, browser will use the HTTP protocol to connect the site. When client initiates HTTP connections, but the server is the HTTPS site, it will return messages of HTTP redirection. The content in this packet contains the actual HTTPS address, such as https://www.yyy.com. The client receives this packet, and browser will be re-launched to initiates HTTPS connection. Compared with directly typing https://www.yyy.com in client browser, this redirection will be no difference.

   Because of the habits of users, they don't pay attention to the difference between HTTP and HTTPS in the URL.

b. **Application in practice :** With HTTP connection, there may be some button on the web-page to initiate HTTPS connections. For example, when user want to login personal account of E-mail and click on the submit button to transmit ID and password, initiates HTTPS connections to protect confidentiality of personal information.

   Because of consideration of the overhead of SSL handshake, only important information has encrypted, not the whole data in the connection. In general, websites don't use HTTPS connection in the whole process, because HTTPS connection is usually 2 to 100 times slower than HTTP connection [29]. Therefore, submission of confidential information (such as ID, password) is by HTTPS connection, and other services are still using HTTP connection. In this

way, the delivery of HTTPS URL is in the HTTP message content, while the inherent insecurity of HTTP protocol, so it results in security vulnerabilities.

Web-based applications rely on the HTTPS protocol to guarantee privacy and security in transactions ranging from home banking, e-commerce, and E-procurement to those that deal with sensitive data such as career and identity information. The process used by HTTPS to ensure data is secure centers around the distribution of certificates between the server, the client, and a trusted third party. This involves a few distinct steps, which are briefly simplified in Figure 2.3

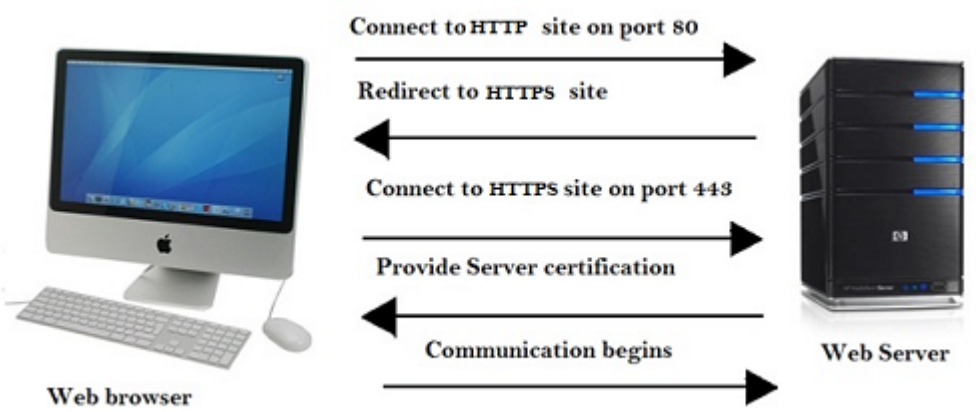


Figure 2.3: The HTTPS Communication Process

The process outlined in Figure 2.3 is by no means detailed, but basically works out as follows:

i. The client browser connects to http://mail.yyy.com on port 80 using HTTP.

ii. The server redirects the client HTTPS version of this site using an HTTP code 302 redirect.

iii. The client connects to https://mail.yyy.com on port 443.

iv. The server provides a certificate to the client containing its digital signature. This certificate is used to verify the identity of the site

v. The client takes this certificate and verifies it against its list of trusted certificate authorities.

vi. Encrypted communication ensues.

If the certificate validation process fails then that means the website has failed to verify its identity. At that point the user is typically presented with a certificate validation error and they can choose to proceed at their own risk, because they may or may not actually be communicating with the website they think they are talking to[30].

2.6 Defeating HTTPS

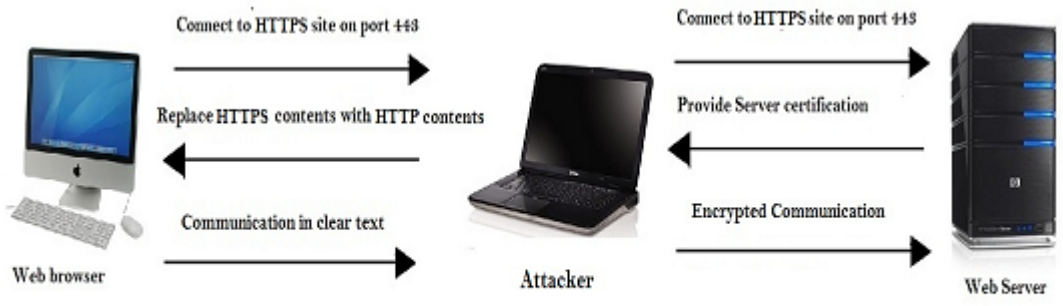


Figure 2.4: Hijacking HTTPS Communication

The process outlined in Figure 2.4 works like this:

i. Traffic between the client and web server is intercepted.

ii. When an HTTPS URL is encountered sslstrip replaces it with an HTTP link and keeps a mapping of the changes.

iii. The attacking machine supplies certificates to the web server and impersonates the client.

iv. Traffic is received back from the secure website and provided back to the client.

Chapter 3

Objectives

i. To study and explorer the various session hijacking attacks.

ii. To find the vulnerability of the operating systems which leads to HTTP session hijacking attack.

iii. To identify prevention mechanism to get rid of HTTP session hijacking attack.

iv. To find the dependency and scope of HTTP session hijacking attack.

Chapter 4

Implementation Details and Experimental Results

All the experiments have been conducted on the wired LAN switching environment. All computers have been connected by using Cisco Catalyst Express 500 switch. All five operating systems Windows xp, Windows 7, Windows 8 consumer preview, CentOS 6.0 and Ubuntu 11.10 have been installed on same type of hardware configuration. The basic configuration of All computer are Intel core 2 due 3.00 Ghz processor and 2 GB RAM (sufficient for each operating system). Since this experiment are conducted to measure the security level of each operating system against the session hijacking attack. To allow session hijacking no anti ARP or static ARP spoofing programs have been conducted on victim's computer. On the gate-way router computer, static ARP and static port[31] have not been conducted as well.

A server, for experimental purpose has been created in Linux. An apache web server has been connected with Mysql server by the means of PHP programming. Myql server contains the data of users, it contains the user-name and password of users. Web server takes the input from the user and matches its user-name and password with database resides in Mysql server by the means of PHP programming. If entered user-name and password are correct, web server creates a session id, on the basis of parameters which have been set in the script. This session id is checked on each page visited by user. Authorized user can access the web pages of server by means of session id. After entering the user-name and password,

generated session id is an only proof of authorized user. This session id remains active until logout. Web-pages are designed in such a way that browser would automatically refresh pages after 10 seconds.

Steps performed during configuration and implementations are

4.1 Server setup

i. To establish a LAMP (Linux Apache Mysql Php) server Redhat 5.0 has been used.

ii. Apache web server has been created by using Virtual host. The configuration file used by apache is /etc/httpd/conf/httpd.conf. It stores information on various functions of the server, which can be edited by removing or adding a number sign # at the beginning of the line, thus setting values for each directive. In order to create web server the lines as shown in Figure 4.1 must be written at the end of httpd.conf file.

```
<VirtualHost 192.168.0.1>
        ServerAdmin     root@ccnalab.com
        DocumentRoot    /var/www/htm
        ServerName      *.ccnalab.com
        ErrorLog        logs/ccna-error_log
        CustomLog       logs/ccna-acess_log     common
</VirtualHost>
```

Figure 4.1: Apache Configuration

192.168.0.1. is the ip address of web server. After configure the web server need to restart the httpd service by typing command "service httpd restart" under root privileges.

iii. Configuration of Mysql server to start the Mysql server immediately, type the following from a terminal window as root user: "service mysqld start". To set the Mysql server to start each time the computer reboots, type the following (as root): "chkconfig mysqld on"

a. Once the mysqld server is running. To add the root user as a Mysql administrator, log in as the root user and type the following from a terminal window (substituting your own password in place of myownpasswd): "# mysqladmin -u root password myownpasswd"

b. A database named "login" has been created by using command "create database login".

c. A table which store user-name and password, has been created under database "login". In order to do this following commands have been used as shown in Figure 4.2 led by "use database login"

```
mysql> create table users( id int not null auto_increment, user varchar(20) not
null, pass varchar(20) not null, unique(user),primary key(id));
Query OK, 0 rows affected (0.02 sec)
```

Figure 4.2: Creating table

```
mysql> insert into users(user,pass) values ('admin','12345678');
Query OK, 1 row affected (0.00 sec)

mysql> exit
Bye
[root@localhost ~]# service mysqld restart
```

Figure 4.3: Inserting values in table

iv. PHP Scripts has been used to connect the webserver to Mysql server

a. Index.php is used for user interface to enter user-name and password.

b. Index.php would pass the user-name and password parameter to login.php after submit the form. Login.php verify the user-name and password with the stored databases in Mysql server.

c. Login.php include a file named connect.php which connects the database of Mysql server to the login.php.

d. If user-name and password matches to the admin user and password then it will open the admin.php.

```
<html>
<body background="wel.jpg">
<font size="50" face="sans-serif">Login</font>
<br>
<form action="login.php" method="POST">
<table-cellpadding="3" cellspacing="4" border="0">
<tr>
<td> Username:</td>
<td><input type="text" name="user" /></td>
</tr>

<tr>
<td> Password:</td>
<td> <input type="password" name="pass" /> </td>
</tr>
<tr>
<td> <input type="Submit" name="submit" value="login" /> </td>

</tr>
</table>
</form>
</body>
</html>
```

Figure 4.4: Index.php user-interface for user to enter the username and password.

e. If user-name and password matches to the normal user and its password then it will open the home.php

f. Logout.php used for to kill the current session of any user. If user call that script The script kill the current session from the web-server.

```php
<?php
session_start();
include_once("connect.php")
?>
<?php
$uname = $_POST['user'];
$pass =  $_POST['pass'];
$sql = "SELECT count(*) FROM users where (
user='".$uname."' and pass='".$pass."')";
$qury = mysql_query($sql);
$result = mysql_fetch_array($qury);
if($result[0]>0)
{
if($uname =="admin")
{
 $_SESSION['userName'] = 'admin';
header("location: admin.php");
//echo"<br /><a href='logout.php'>LOGOUT</a>";
}
else
{
$_SESSION['userName'] = $uname;
header("location: home.php");
}
}
else
{
echo "Login Failed";
}
?>
```

Figure 4.5: login.php verify user-name and password

```php
<?php

$connect = mysql_connect("localhost","root", "123456") or die
(mysql_error());

mysql_select_db("login") or die(mysql_error());

?>
```

Figure 4.6: Connect.php

```
<head>

<meta http-equiv="refresh" content="10" />
</head>

<?php
session_start();

if ($_SESSION['userName']=='admin')
{
echo "THis is admin page";
echo "<br />You were already logged in ".$_SESSION
['userName']."."";

echo"<br /><a href='logout.php'>LOGOUT</a>";
echo"<br /><a href='ins.php'>INSERT Record</a>";
echo"<br /><a href='display.php'>To display Record</a>";

}
else
{
header("location: index.php");
}
?>
```

Figure 4.7: Admin.php

```
<head>

<meta http-equiv="refresh" content="10" />
</head>


<?php
session_start();

if (isset($_SESSION['userName']))
{
echo "<br />THis is Home page Welcome $userName";
echo "<br />You were already logged in ".$_SESSION
['userName']."."";

echo"<br /><a href='logout.php'>LOGOUT</a>";


}
else
{
header("location: index.php");
}
?>
```

Figure 4.8: home.php for normal users.

```
 <?php

include_once("connect.php"); ?>
<?php
session_start();

session_unset();

session_destroy();

if($_SESSION['userName']=="")
{
echo "Successfully logger out!<br/>";
header("location: index.php");
}
else
{
echo "Error Occured !!<br />";
}
?>
```

Figure 4.9: Logout.php for kill the session

4.2 Client Side

After each modification in webserver there is need to restart the service of httpd by using command "service httpd restart". Admin.php and home.php are designed in such a way that browser would automatically refresh pages after 10 seconds.

When client open the web address 192.168.0.1 on the web browser. It will open the index.php as shown in Figure 4.10.

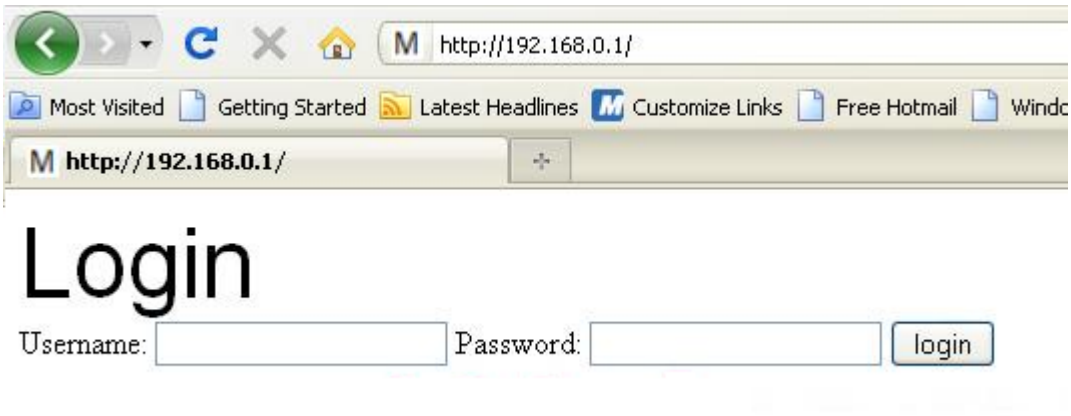


Figure 4.10: index.php Open in web-browser

If admin user-name and its password is entered then admin page would be displayed as shown in Figure 4.11.

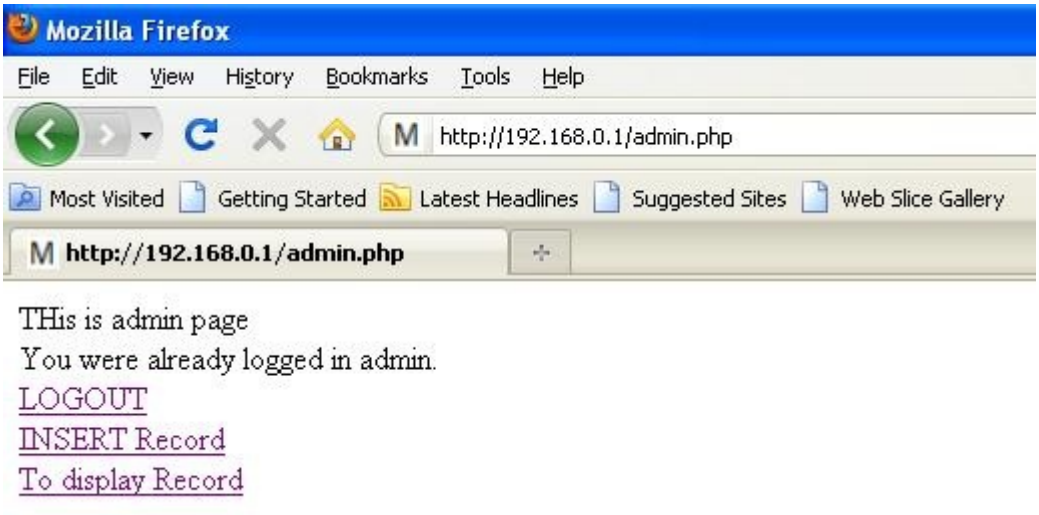


Figure 4.11: admin.php open in web browser

Each operating system has been installed on different computer system. Tools installed on attacker machine. Wireshark for passive sniffing[32]. Cain and abel for active sniffing[33]. Sessionthief for cookie stealing[34]. Cookie injector script installed on google chrome[35].

4.3 Session Hijacking by sniffing the cookie

In order to perform session hijacking three steps have to be performed.

i. Cookie Sniffing.

ii. Cookie stealing.

iii. Cookie injection.

Step i. Use Sniffing to capture the traffic of victim. If computers are connected in hub environment use passive sniffing to capture the HTTP packet of victim. Figure 4.12 shows that cookie captured by wireshark.

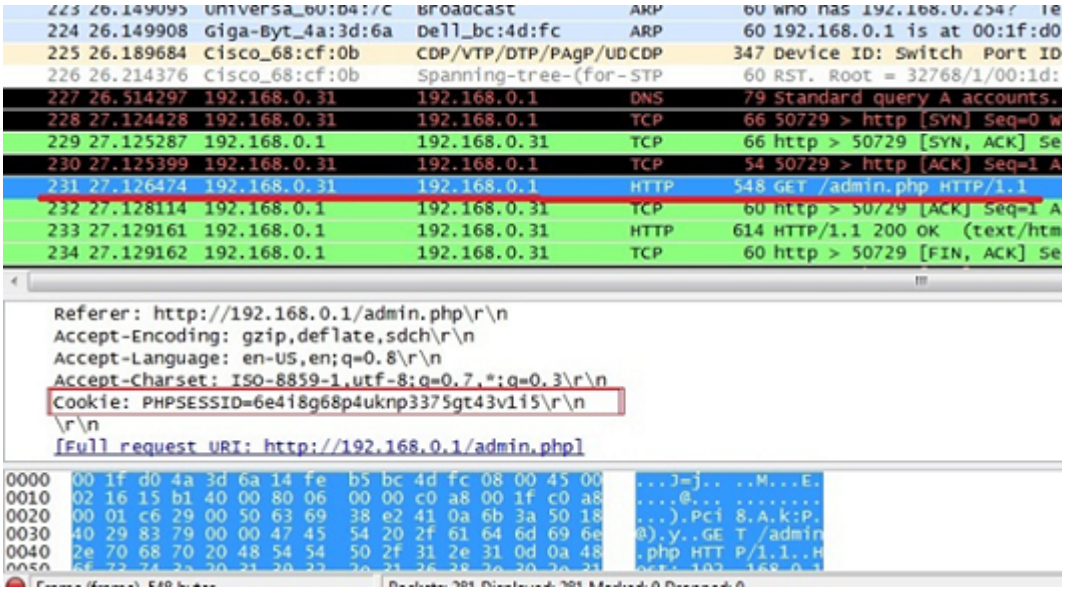


Figure 4.12: cookie captured by passive sniffing

If computer are connected in switch environment use active sniffing to capture the HTTP packet of victim. Active sniffing or arp spoofing can be done by cain & abel tool. For cookie capturing session thief tool can be used as show in Figure 4.13.

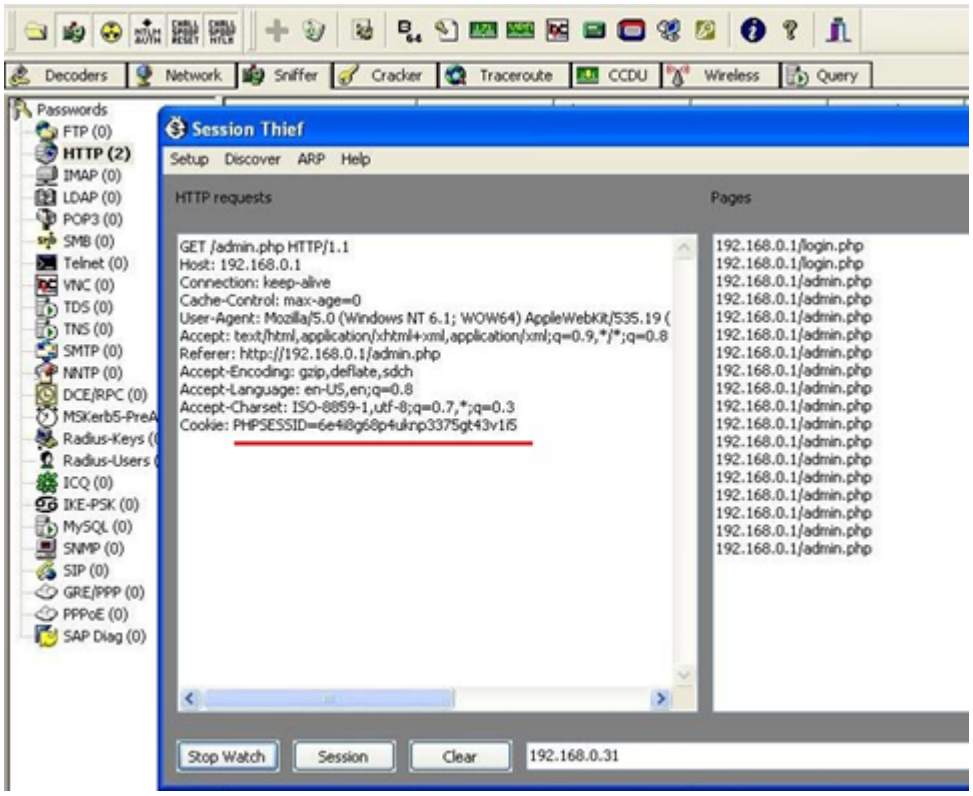


Figure 4.13: cookie captured by session thief

Step ii. Copy cookie value and open the web-browser. Cookie injector script should be installed in the web-browser. Press button Alt+N, a cookie injector field would be displayed and paste the cookie value in that field as shown in Figure 4.14.

Step iii. Open the page http://192.168.0.1/admin.php or http://192.168.0.1/home.php . It open the admin page without asking user-name and password as shown in Figure 4.15.

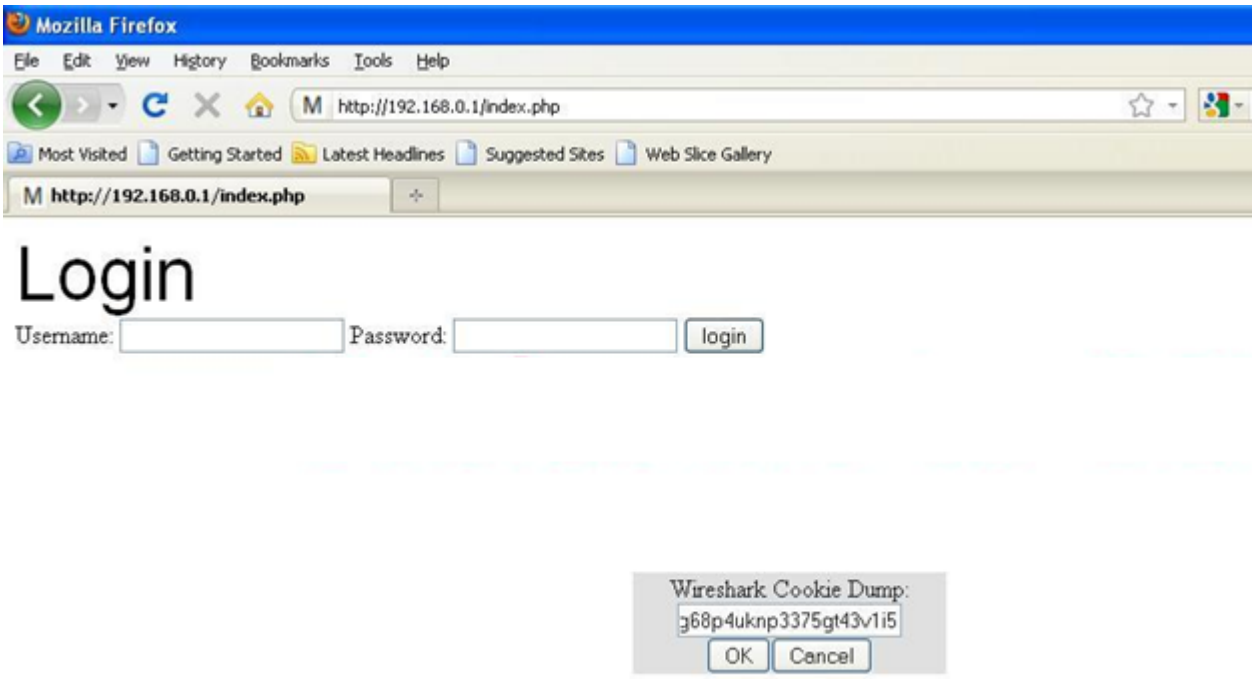


Figure 4.14: cookie injection

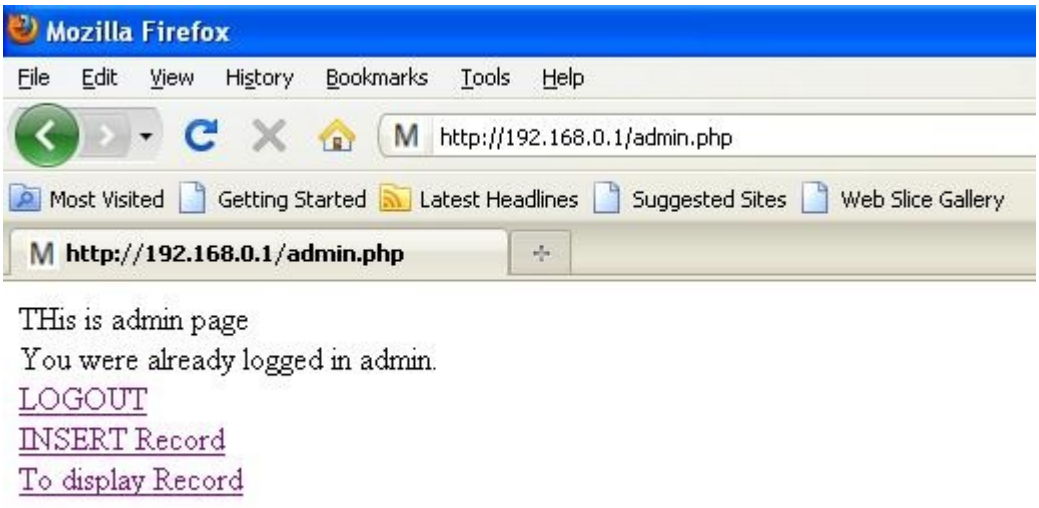


Figure 4.15: Accessing admin page by cookie injection

4.4 Passive sniffing with/without firewall

When no firewall was enabled on all operating system. IP configurations of all operating systems

Table 4.1: IP configurations of all operating systems

Operating System	IP Address
Redhat 5.0(server)	192.168.0.01
Ubuntu 11.10	192.168.0.27
CentOS 6.0	192.168.0.32
Windows xp sp 2	192.168.0.29
Windows 7 Ultimate	192.168.0.21
Windows 8 Consumer Preview	192.168.0.08

Admin account was opened on all operating systems. Passive sniffing was used to capture the broadcast packets. In order to do passive sniffing wireshark tool was used. The aim of passive sniffing is to get session cookie value. Session cookie value resides in HTTP packet. Passive sniffing checks that which operating systems broadcast such a packet which leads to session hijacking. After the experiment, the results of passive sniffing are as follows. If cookie is captured the result would be recorded as "Yes" otherwise result would be recorded as "No". First Experiment has been done when no firewall was enabled on all operating systems.

Table 4.2: Passive sniffing with/without firewall

Operating System	Server	Result
Ubuntu 11.10	Redhat 5.0	No
CentOS 6.0	Redhat 5.0	No
Windows xp sp2	Redhat 5.0	No
Windows 7 ultimate	Redhat 5.0	No
Windows 8 Consumer preview	Redhat 5.0	No

After sniffing it was found that windows xp, windows 7 and windows 8 send more packets in network but nothing is vulnerable to session hijacking. Figure 4.16 shows that Ubuntu 11.10 broadcast only one packet. Without firewall not even

single operating system is vulnerable to session hijacking under passive sniffing. So there is no need to set any rule in firewall(s).

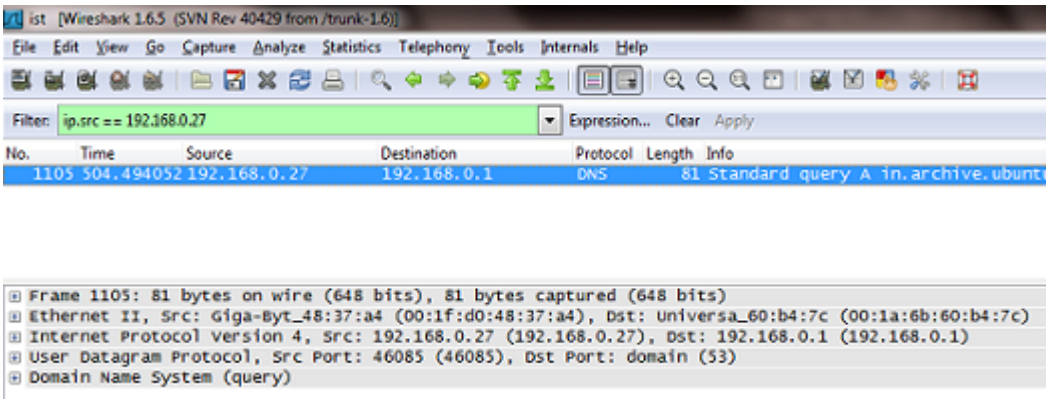


Figure 4.16: Ubuntu broadcast

Ubuntu broadcasted only one packet during sniffing as shown in Figure 4.16

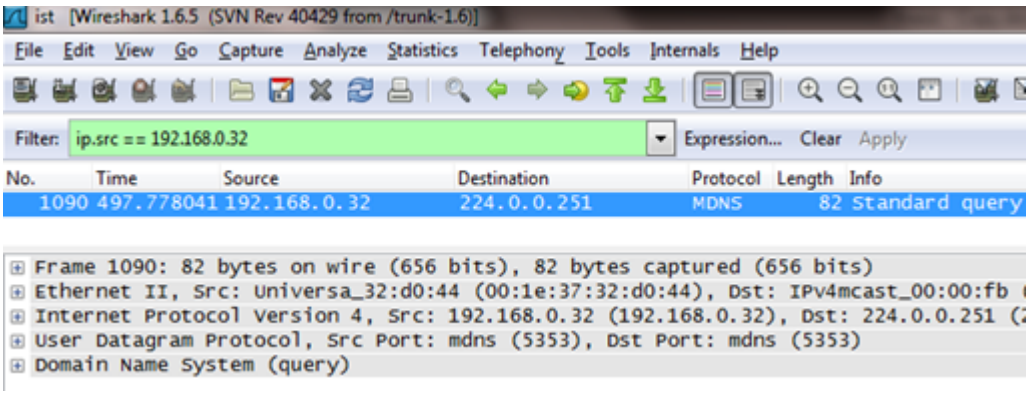


Figure 4.17: CentOS broadcast

Centos broadcasted only one packet during sniffing as shown in Figure 4.17. Both linux Ubuntu and Centos broadcasted same type of packet

Windows xp broadcasted ICMP and NBNS (NetBIOS Name Service) packets as shown in Figure 4.18. NBNS serves much the same purpose as DNS does: translate human-readable names to IP addresses NetBIOS services translates the NetBIOS names to IP addresses

Windows 7 Ultimate broadcasted NBNS (NetBIOS Name Service) packets as shown in Figure 4.19.

Windows 8 Consumer Preview broadcasted LLMNR (Link Local Multicast Name Resolution) ,SSDP (Simple Service Discovery Protocol) and NBNS (NetBIOS Name Service) packets as shown in Figure 4.20

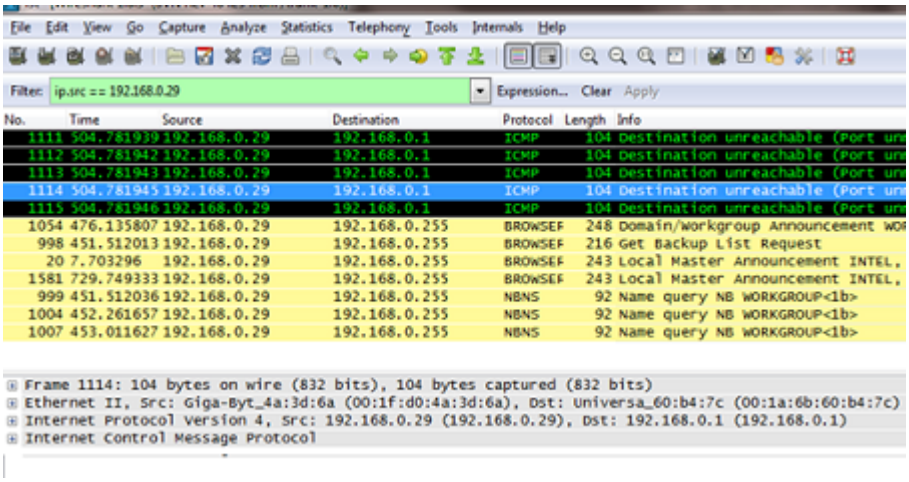


Figure 4.18: Windows xp sp 2 broadcast

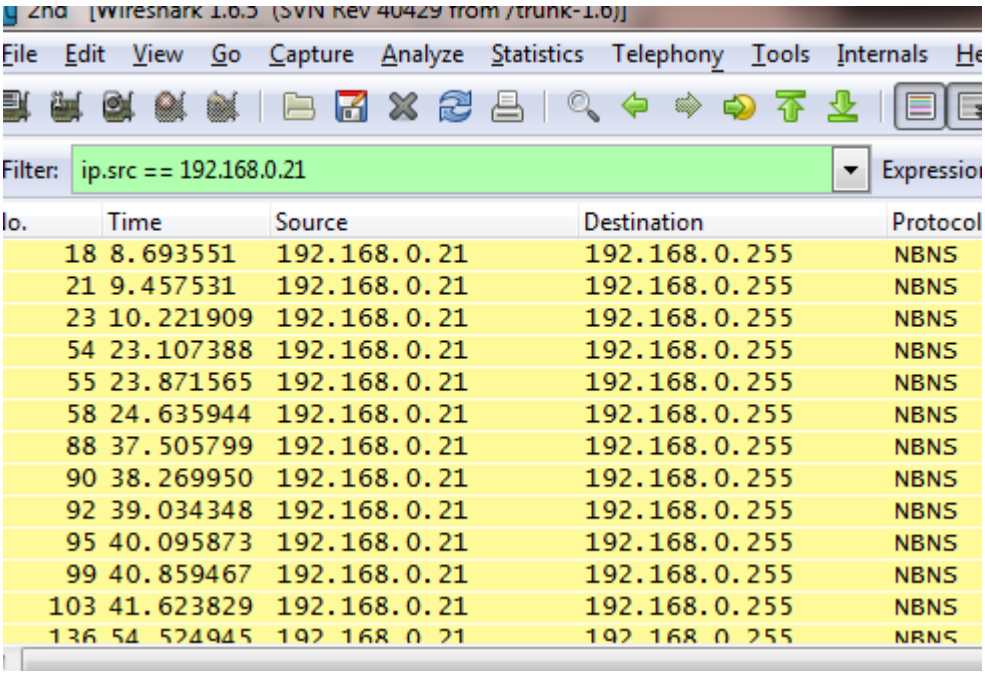
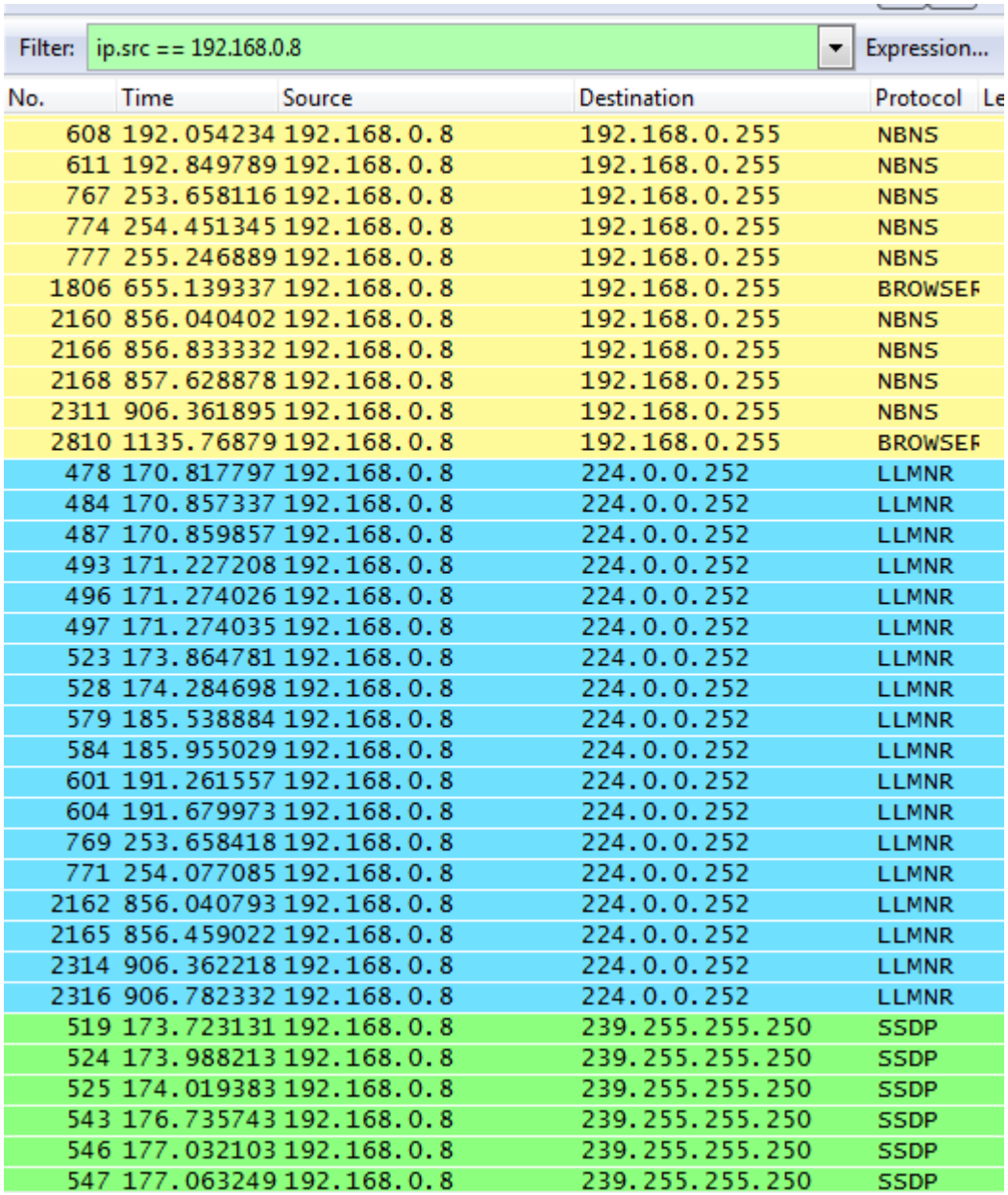


Figure 4.19: Windows 7 Ultimate broadcast



No.	Time	Source	Destination	Protocol	Le
608	192.054234	192.168.0.8	192.168.0.255	NBNS	
611	192.849789	192.168.0.8	192.168.0.255	NBNS	
767	253.658116	192.168.0.8	192.168.0.255	NBNS	
774	254.451345	192.168.0.8	192.168.0.255	NBNS	
777	255.246889	192.168.0.8	192.168.0.255	NBNS	
1806	655.139337	192.168.0.8	192.168.0.255	BROWSEF	
2160	856.040402	192.168.0.8	192.168.0.255	NBNS	
2166	856.833332	192.168.0.8	192.168.0.255	NBNS	
2168	857.628878	192.168.0.8	192.168.0.255	NBNS	
2311	906.361895	192.168.0.8	192.168.0.255	NBNS	
2810	1135.76879	192.168.0.8	192.168.0.255	BROWSEF	
478	170.817797	192.168.0.8	224.0.0.252	LLMNR	
484	170.857337	192.168.0.8	224.0.0.252	LLMNR	
487	170.859857	192.168.0.8	224.0.0.252	LLMNR	
493	171.227208	192.168.0.8	224.0.0.252	LLMNR	
496	171.274026	192.168.0.8	224.0.0.252	LLMNR	
497	171.274035	192.168.0.8	224.0.0.252	LLMNR	
523	173.864781	192.168.0.8	224.0.0.252	LLMNR	
528	174.284698	192.168.0.8	224.0.0.252	LLMNR	
579	185.538884	192.168.0.8	224.0.0.252	LLMNR	
584	185.955029	192.168.0.8	224.0.0.252	LLMNR	
601	191.261557	192.168.0.8	224.0.0.252	LLMNR	
604	191.679973	192.168.0.8	224.0.0.252	LLMNR	
769	253.658418	192.168.0.8	224.0.0.252	LLMNR	
771	254.077085	192.168.0.8	224.0.0.252	LLMNR	
2162	856.040793	192.168.0.8	224.0.0.252	LLMNR	
2165	856.459022	192.168.0.8	224.0.0.252	LLMNR	
2314	906.362218	192.168.0.8	224.0.0.252	LLMNR	
2316	906.782332	192.168.0.8	224.0.0.252	LLMNR	
519	173.723131	192.168.0.8	239.255.255.250	SSDP	
524	173.988213	192.168.0.8	239.255.255.250	SSDP	
525	174.019383	192.168.0.8	239.255.255.250	SSDP	
543	176.735743	192.168.0.8	239.255.255.250	SSDP	
546	177.032103	192.168.0.8	239.255.255.250	SSDP	
547	177.063249	192.168.0.8	239.255.255.250	SSDP	

Figure 4.20: Windows 8 Consumer Preview broadcast

4.5 Active sniffing with/without firewall

When no firewall was enabled and active sniffing (ARP spoofing) had been conducted on all operating systems by attacker's machine. The results of active sniffing are as follows.

Table 4.3: Active sniffing with/without firewall

Operating System	Server	Result
Ubuntu 11.10	Redhat 5.0	Yes
CentOS 6.0	Redhat 5.0	Yes
Windows xp sp2	Redhat 5.0	Yes
Windows 7 ultimate	Redhat 5.0	Yes
Windows 8 Consumer preview	Redhat 5.0	Yes

After sniffing it was found that cookie can be stolen by using ARP spoofing. This shows that HTTP session hijacking attack directly depend upon the ARP spoofing. When firewalls of Windows and Linux (Centos and Ubuntu) are enabled, they offer many rules regarding inbound and outbound traffic but there is no any rule which can avoid ARP spoofing. So when firewalls are enabled still ARP spoofing produced same result as shown in Table 4.3 because there is no any rule in firewalls which can restrict the ARP spoofing. ARP spoofing can be avoided by static entry in ARP cache but this is not good solution. ARP spoofing can be detected by some free software. In Linux environment a software named ARPwatcher can detect ARP spoofing. Service of Arp watcher can be started by typing command

"Service arpwatch restart"

Arpwatch can be binded with interface eth0 by using command

"arpwatch -i eth0". By default, arpwatch send its log details to /var/log/messages. To monitor all arpwatch log messages

tail -f /var/log/messages |grep arpwatch

cat /var/log/messages |grep arpwatch

If arp spoofing happens then there is a flip flop in the file as shown in the Figure 4.21

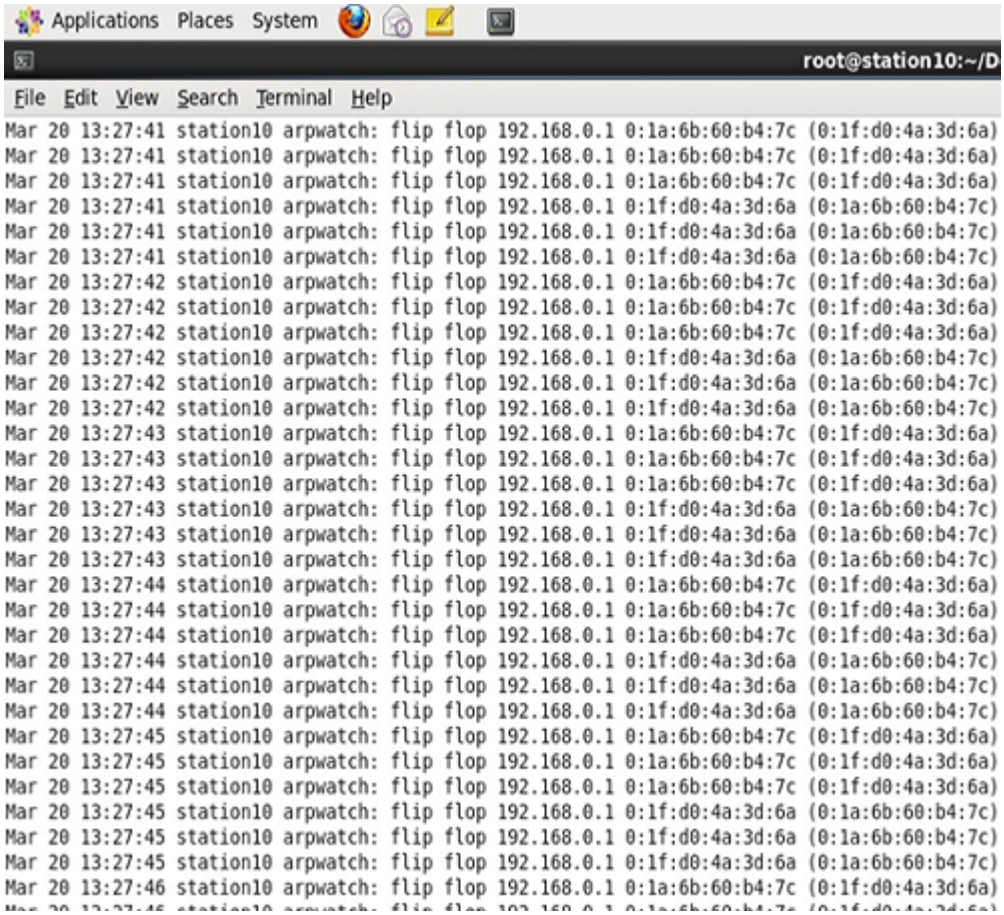


Figure 4.21: Linux ARP watch log file

In windows environment free software DecaffeinatID act as ARP watcher it maintains a log file as well gives a special alert as shown in Figure 4.22 whenever it sees the MAC address of the gateway change.

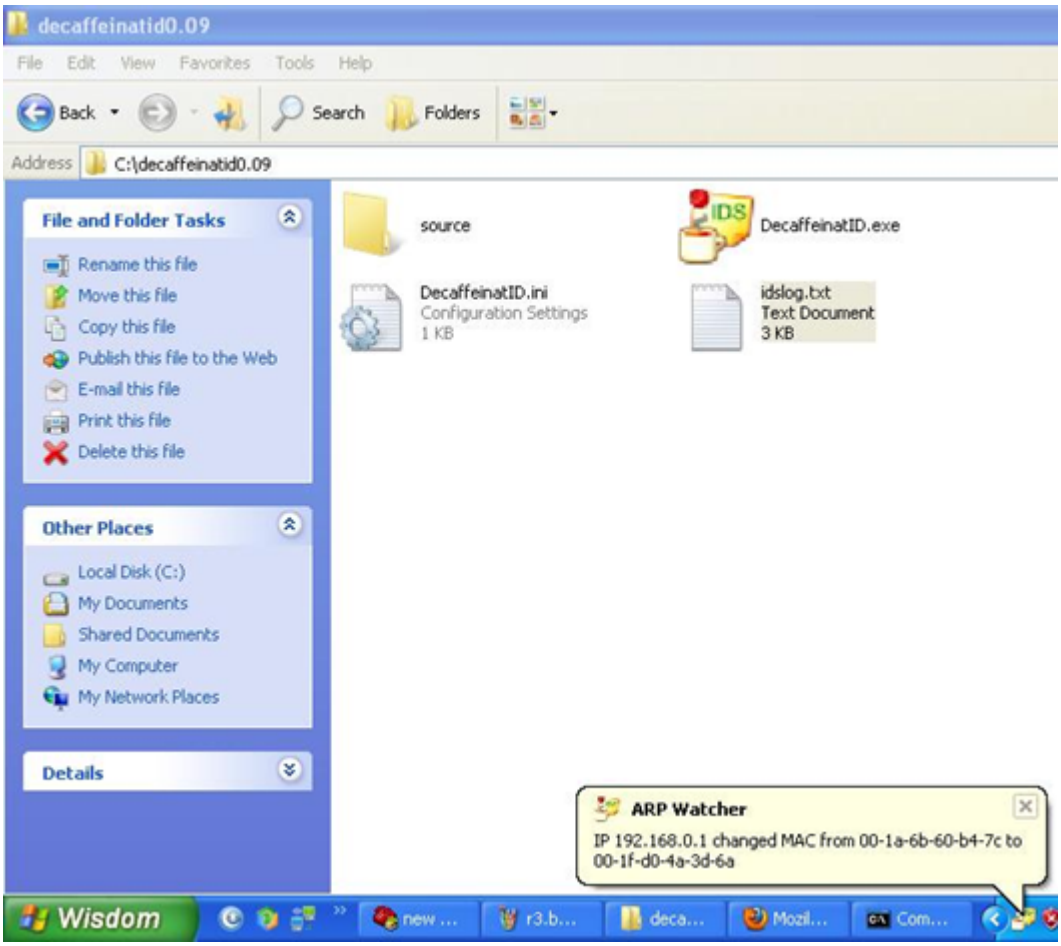


Figure 4.22: Special Alert by DecaffeinatID

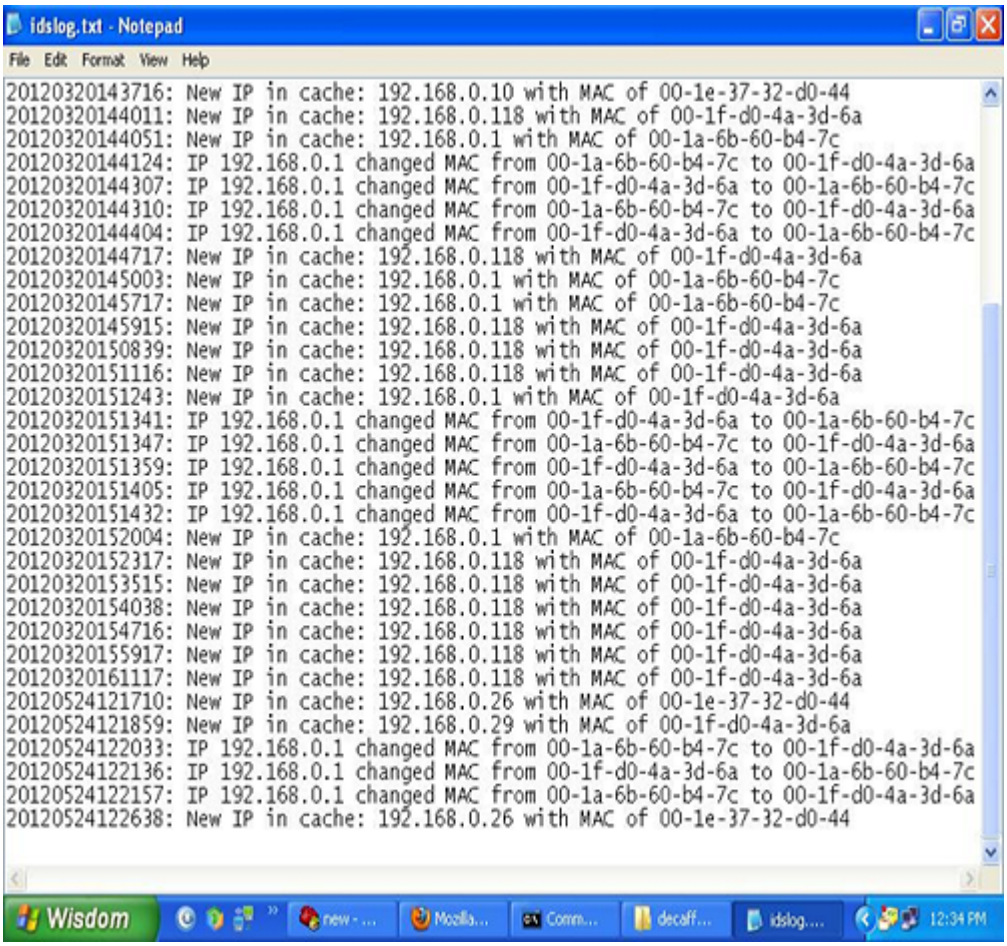


Figure 4.23: Log file of DecaffeinatID

4.6 Base and Limitations of session hijacking

So far session hijacking depend upon the cookie stealing. Cookie stealing can be done only in one broadcast domain or in one subnet. So session hijacking attack is limited only in one broadcast domain. But if weak algorithm or weak parameters have been applied in programming in order to create session id of user then session id can be guessed. When victim logged in and attacker guess the exact session id of victim then victim's session can be hijacked from anywhere or not. The base of this experiment is that the HTTP session hijacking can be done or not from anywhere if it depends upon the cookie guessing. Figure 4.24 showing a topology of the experimental setup in order to verify, can session hijacking be done from anywhere if it depends upon cookie guessing.

4.6.1 Implementation

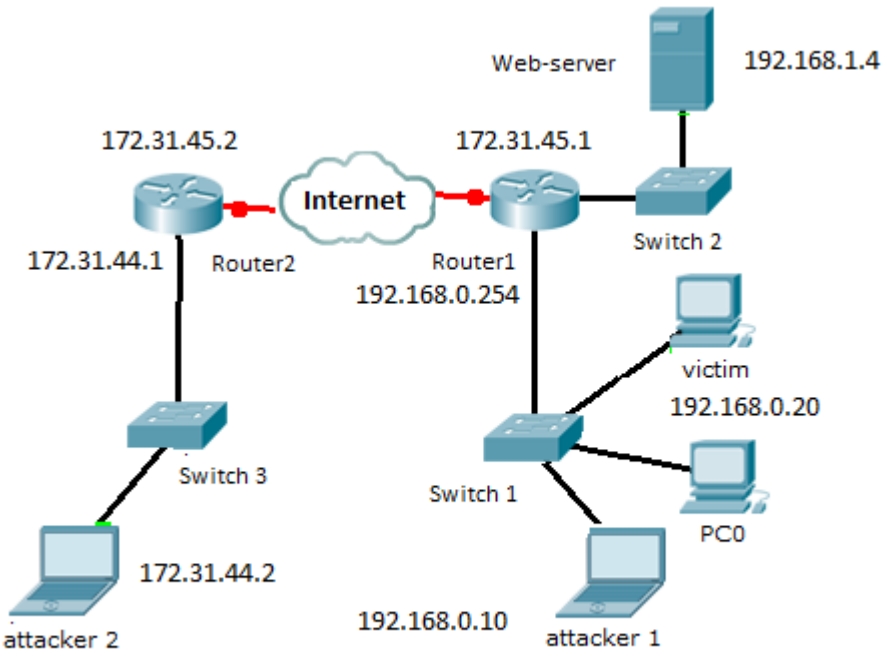


Figure 4.24: Router implementation

The router configuration of router1 is shown in Figure 4.25

Similarly router2 configuration shown in Figure 4.27

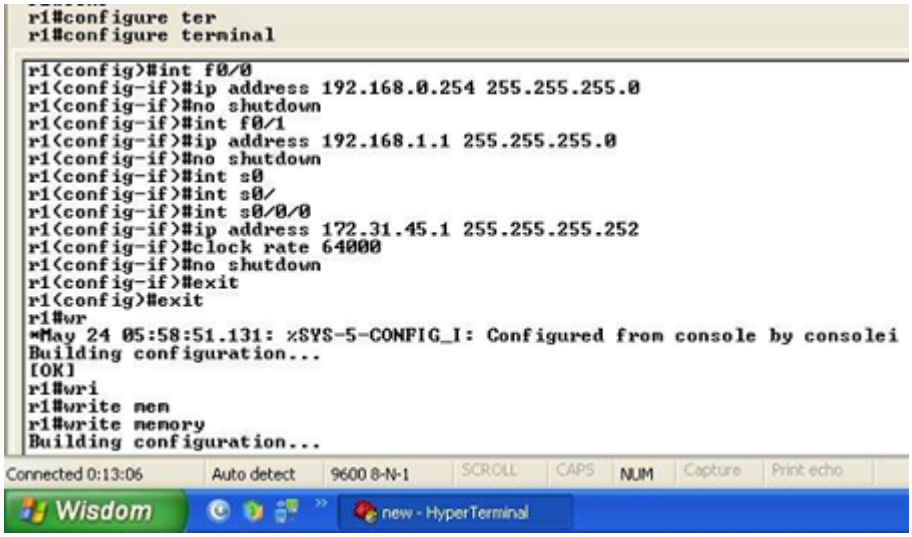


Figure 4.25: IP addresses of interfaces of router1

```
r1#configure t
r1#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
r1(config)#ip route 172.31.44.0 255.255.255.0 s0/0/0
r1(config)#
r1(config)#
r1(config)#
```

Figure 4.26: Static Routing on Router1

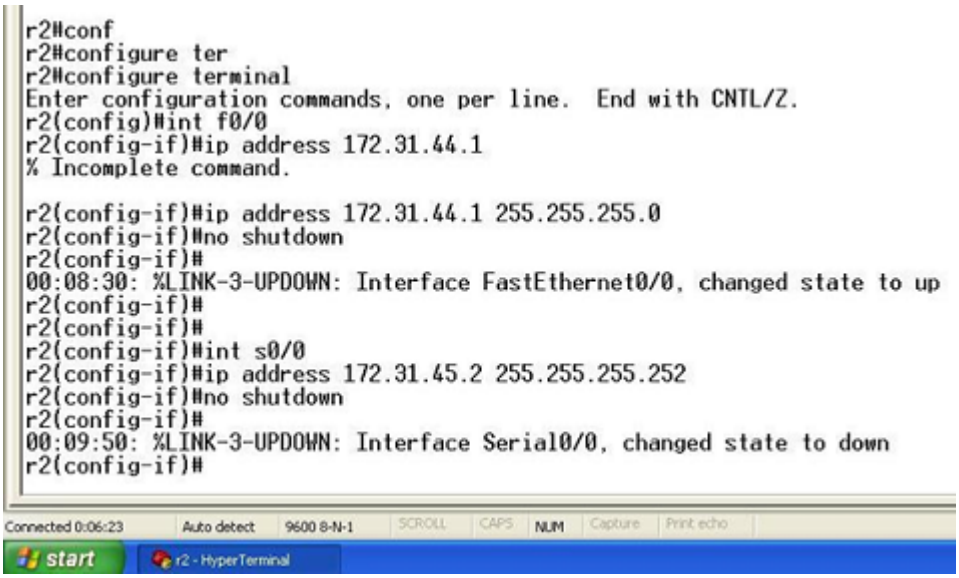


Figure 4.27: IP addresses of interfaces of router2

After experiment it was found if attacker2 guesses exact session id of victim pc then session hijacking can be performed. Session hijacking can be done from anywhere if it depend upon cookie guessing.

```
r2#conf t
r2#conf terminal
Enter configuration commands, one per line.  End with CNTL/Z.
r2(config)#ip route 192.168.0.0 255.255.255.0 s0/0
r2(config)#ip route 192.168.1.0 255.255.255.0 s0/0
r2(config)#exit
r2#
r2#
r2#
00:20:53: %SYS-5-CONFIG_I: Configured from console by console
r2#
r2#
```

Figure 4.28: Static Routing on Router2

Chapter 5

Conclusion and Future Scope

Session hijacking attack can be done by taking the advantage of the vulnerabilities of network, operating systems and web-accounts. Session hijacking attack does not need the password to gaining access of web-accounts of victims. In this thesis the work have been done to know the vulnerabilities of different operating systems. No vulnerability has been found in operating systems when passive sniffing on switching environment is enabled. When active sniffing is used then session hijacking can be performed. Microsoft Windows inbuilt firewall and linux IPTables firewall contain no rule to avoid active sniffing. But there are some free tools like Arp watcher in Linux systems and DecaffeinatID in Windows systems which can detect the arp spoofing. It has been found that if HTTP session hijacking depend upon the cookie stealing then it can be done only in one broadcast domain but if it depend upon the cookie guessing then it can be done from anywhere on network.

In future, work can be extended to include other existing or forthcoming operating system like Apple's OS X Lion, Microsoft windows 8 full version, new version of Ubuntu and SUSE linux.

References

[1] "Greenlaw R and Hepp E", Inline/Online: Fundamentals of the Internet and the World Wide Web, 2/e McGraw-Hill Higher education, 2005.

[2] "Proactive vs. Reactive Security",http://www.crime-research.org/library/Richard.html, 2009.

[3] http://en.wikipedia.org/wiki/Promiscuous_mode.

[4] http://en.wikipedia.org/wiki/Denial-of-service_attack.

[5] http://en.wikipedia.org/wiki/Phishing.

[6] Goodchild Joan ,(11 January 2010)., "Social Engineering: The Basics," Online Available http://www.csoonline.com/article/514063/social-engineering-the-basics.

[7] Jie Qin, Huijuan Yan, Qun Si, Fuliang Yan,"A Trojan horse Detection Technology Based on Behavior Analysis", Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference

[8] McFedries, P. (Aug. 2005), Technically Speaking: The Spyware Nightmare, IEEE Spectrum, Vol. 42, Iss. 8, p. 72-72

[9] http://en.wikipedia.org/wiki/Computer_virus

[10] Shigang Chen and Sanjay Ranka, "Detecting Internet Worms at Early Stage", IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 23, NO. 10, OCTOBER 2005.

[11] Laurianne McLaughlin, "Bot Software Spreads, Causes New Worries", IEEE DISTRIBUTED SYSTEMS ONLINE 1541-4922 © 2004 Published by the IEEE Computer Society Vol. 5, No. 6; June 2004.

[12] Crispin Cowan, Perry Wagle, Calton Pu, Steve Beattie, and Jonathan Walpole, "Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade", DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings.

[13] Robert T. Morris, "A Weakness in the 4.2BSD Unix TCP/IP Software", AT&T Bell Laboratories Murray Hill, New Jersey 07974, February 1985

[14] http://www.codeproject.com/Articles/298822/Session-Hijacking

[15] Mitja Kolsek, "Session Fixation Vulnerability in Web-based Applications", December 2002 (Revised February 2007 - the Acknowledgments section), Available at http://www.acrossecurity.com/papers/session_fixation.pdf

[16] "Cross Site Scripting Attack", http://www.acunetix.com/websitesecurity/cross-site-scripting.htm

[17] David Endler. "The evolution of cross-site scripting attacks", Whitepaper, iDefense Inc., 20. May 2002.

[18] "PHP Session Variables", http://www.w3schools.com/php/php_sessions.asp

[19] http://php.net/manual/en/reserved.variables.session.php

[20] Satoshi Munakata and Masahiro Hiji, "A Session Management Method to Improve Web Applications Usability on Mobile Network" IEEE 8th International Conference on Computer and Information Technology CIT 2008 (2008)

[21] Kennedy, S. (2004), "Best practices for wireless network security" Information Systems Control Journal (3).

[22] Adam Stubblefield, John Ioannidis, and Aviel D. Rubin, "Using Fluhrer, Mantin, and Shamir Attack to Break WEP", AT&T Labs Technical Report 2001.

[23] Mobile Antivirus Researcher's Association, "The Ten Most Critical Wireless and Mobile Security Vulnerabilities" http://www.net-security.org/article.php?id=927, Thursday, 29 June 2006.

[24] "Hacking Techniques in Wireless Networks", http://www.cs.wright.edu/~pmateti/InternetSecurity/Lectures/WirelessHacks/Mateti-WirelessHacks.htm

[25] http://www.kismetwireless.net.

[26] Preecha Noiumkar, Thawatchai Chomsiri, "Top 10 Free Web-Mail Security Test Using Session Hijacking",Third 2008 International Conference on Convergence and Hybrid Information Technology.

[27] Marlingspike Moixe, "New Tricks For Defeating SSL in Practice", BlackHat Conference, USA(2009).

[28] Peter Burkholder, "SSL Man-in-the-Middle Attacks", SANS Institue InfoSec Reading, 2003.

[29] S.Thomas, "SSL and TLS Essentials", New york: Wiley Computing Publishing, 2004.

[30] Michael Howard, "Man-in-the-Middle Attack to the HTTPS Protocol", IEEE computer society, 2009, pp.78-81.

[31] "Cisco IOS Switch Security Configuration Guide",www.cisco.com

[32] http://www.wireshark.org/

[33] http://www.oxid.it/cain.html

[34] http://www.rootrulerz.com/2010/12/http-session-hijacking-using-session.html

[35] http://userscripts.org/scripts/show/109320

List of Publications

Mohit, Maninder Singh "Comparative Analysis of Session Hijacking on Different Operating System", JOURNAL OF COMPUTER TECHNOLOGY & APPLICATIONS. ISSN: 2229 - 6964 (communicated).