FALL SEMESTER 2021-2022

CSE3501-Information Security Analysis and Audit

Digital Assignment -1

Submitted by: Alokam Nikhitha

Reg No:19BCE2555

Session hijacking attack

1. INTRODUCTION

Session hijacking, also known as TCP session hijacking, is a method of taking over a web user session by surreptitiously obtaining the session ID and masquerading as the authorized user. Once the user's session ID has been accessed, the attacker can masquerade as that user and do anything the user is authorized to do on the network.

One of the most valuable byproducts of this type of attack is the ability to gain access to a server without having to authenticate to it. Once the attacker hijacks a session, they no longer have to worry about authenticating to the server as long as the communication session remains active. The attacker enjoys the same server access as the compromised user because the user has already authenticated to the server prior to the attack.

What is a session?

HTTP is stateless, so application designers had to develop a way to track the state between multiple connections from the same user, instead of requesting the user to authenticate upon each click in a web application. A session is a series of interactions between two communication end points that occurs during the span of a single connection. When a user logs into an application, a session is created on the server in order to maintain the state for other requests originating from the same user.

Applications use sessions to store parameters that are relevant to the user. The session is kept "alive" on the server as long as the user is logged on to the system. The session is destroyed when the user logs-out from the system or after a predefined period of inactivity. When the session is destroyed, the user's data should also be deleted from the allocated memory space.

A session ID is an identification string (usually a long, random, alpha-numeric string) that is transmitted between the client and the server. Session IDs are commonly stored in cookies, URLs and hidden fields of web pages.

Besides the useful functionality of session IDs, there are several security problems associated with them. Many of the popular websites use algorithms based on easily predictable variables, such as time or IP address, in order to generate the Session IDs, causing their session IDs to be predictable. If encryption is not used (typically SSL), Session IDs are transmitted in the clear and are susceptible to eavesdropping.

2.How does session hijacking work?

The most popular culprits for carrying out a session hijacking are session sniffing, predictable session token ID, man in the browser, cross-site scripting, session sidejacking, session fixation.

2.1 Session sniffing

This is one of the most basic techniques used with application-layer session hijacking. The attacker uses a sniffer, such as Wireshark, or a proxy, such as OWASP Zed, to capture network traffic containing the session ID between a website and a client. Once the attacker captures this value, he can use this valid token to gain unauthorized access.

2.2 Predictable sessions token ID

Many web servers use a custom algorithm or predefined pattern to generate session IDs. The greater the predictability of a session token, the weaker it is and the easier it is to predict. If the attacker can capture several IDs and analyze the pattern, he may be able to predict a valid session ID.

2.3 Man-in-the-browser attack.

This is similar to a man-in-the-middle attack, but the attacker must first infect the victim's computer with a Trojan through some form of trickery or deceit. Once the victim is tricked into installing malware onto the system, the malware waits for the victim to visit a targeted site. The man-in-the-browser malware can invisibly modify transaction information and it can also create additional transactions without the user knowing. Because the requests are initiated from the victim's computer, it is very difficult for the web service to detect that the requests are fake.

2.4 Cross-site scripting

Cybercriminals exploit server or application vulnerabilities to inject client-side scripts into web pages. This causes the browser to execute arbitrary code when it loads a compromised page. If HttpOnly isn’t set in session cookies, cybercriminals can gain access to the session key through injected scripts, giving them the information they need for session hijacking.

2.5 Session side jacking

Cybercriminals can use packet sniffing to monitor a victim’s network traffic and intercept session cookies after the user has authenticated on the server. If TLS encryption is only used for login pages and not for the entire session, cybercriminals can hijack the session, act as the user within the targeted web application.

2.6 Session fixation attacks

This technique steals a valid session ID that has yet to be authenticated. Then, the attacker tries to trick the user into authenticating with this ID. Once authenticated, the attacker now has access to the victim's computer. Session fixation explores a limitation in the way the web application manages a session ID. Three common variations exist session tokens are hidden in an URL argument, session tokens hidden in a form field and session tokens hidden in a session cookie.

The session hijack attack is very stealthy. Session hijack attacks are usually waged against busy networks with a high number of active communication sessions. The high network utilization not only provides the attacker with a large number of sessions to exploit, but it can also provide the attacker with a shroud of protection due to a large number of active sessions on the server.

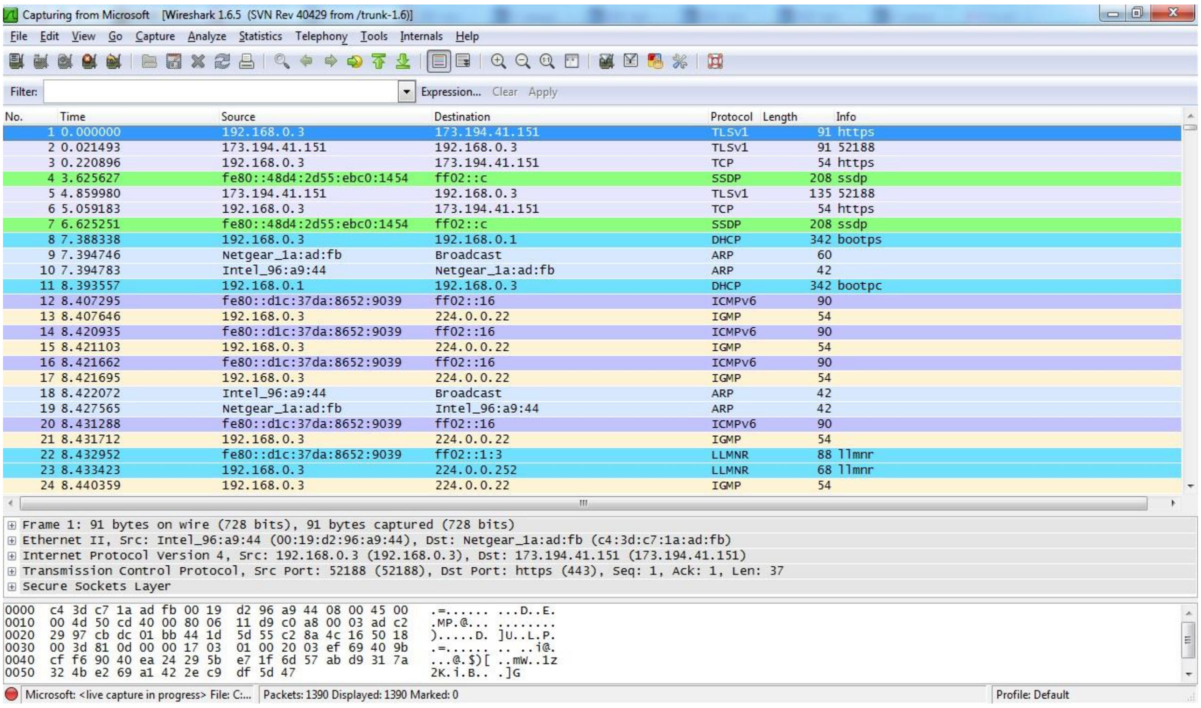
3.Types of session hijacking attacks

There are three types of Session Hijacking attacks:

- **Active Session Hijacking**
- **Passive Session Hijacking**
- **Hybrid Session Hijacking**

3.1 Active session hijacking:

An active session hijacking is the one in which the attacker takes control over an active session of the victim and starts to masquerade as a genuine user by communicating to the server. There are several methods to drop a user‟s connection to the server, one of the most common is to flood the targets machine with huge amount of traffic, and this type of attack is known as Denial of Service. By doing this the attacker puts the user into offline mode, now the attacker has full control over the session. Throughout this process the attacker is in stealth mode listening and monitoring the packets traversing over the network using a packet sniffing tools e.g. wireshark, ethereal, etc.


Capturing from Microsoft [Wireshark 1.6.5 (SVN Rev 40429 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply
Profile: Default

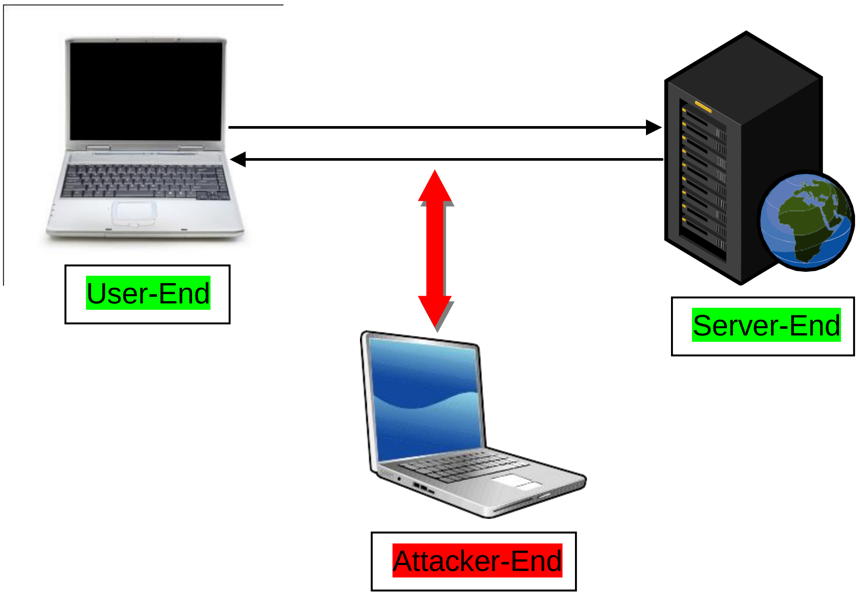
```
No.   Time        Source                      Destination         Protocol Length Info
    1 0.000000    192.168.0.3                 173.194.41.151      TLSv1       91 https
    2 0.021493    173.194.41.151              192.168.0.3         TLSv1       91 52188
    3 0.220896    192.168.0.3                 173.194.41.151      TCP         54 https
    4 3.625627    fe80::48d4:2d55:ebc0:1454   ff02::c             SSDP       208 ssdp
    5 4.859980    173.194.41.151              192.168.0.3         TLSv1      135 52188
    6 5.059183    192.168.0.3                 173.194.41.151      TCP         54 https
    7 6.625251    fe80::48d4:2d55:ebc0:1454   ff02::c             SSDP       208 ssdp
    8 7.388338    192.168.0.3                 192.168.0.1         DHCP       342 bootps
    9 7.394746    Netgear_1a:ad:fb            Broadcast           ARP         60
   10 7.394783    Intel_96:a9:44              Netgear_1a:ad:fb    ARP         42
   11 8.393557    192.168.0.1                 192.168.0.3         DHCP       342 bootpc
   12 8.407295    fe80::d1c:37da:8652:9039    ff02::16            ICMPv6      90
   13 8.407646    192.168.0.3                 224.0.0.22          IGMP        54
   14 8.420935    fe80::d1c:37da:8652:9039    ff02::16            ICMPv6      90
   15 8.421103    192.168.0.3                 224.0.0.22          IGMP        54
   16 8.421662    fe80::d1c:37da:8652:9039    ff02::16            ICMPv6      90
   17 8.421695    192.168.0.3                 224.0.0.22          IGMP        54
   18 8.422072    Intel_96:a9:44              Broadcast           ARP         42
   19 8.427565    Netgear_1a:ad:fb            Intel_96:a9:44      ARP         42
   20 8.431288    fe80::d1c:37da:8652:9039    ff02::16            ICMPv6      90
   21 8.431712    192.168.0.3                 224.0.0.22          IGMP        54
   22 8.432952    fe80::d1c:37da:8652:9039    ff02::1:3           LLMNR       88 llmnr
   23 8.433423    192.168.0.3                 224.0.0.252         LLMNR       68 llmnr
   24 8.440359    192.168.0.3                 224.0.0.22          IGMP        54

⊞ Frame 1: 91 bytes on wire (728 bits), 91 bytes captured (728 bits)
⊞ Ethernet II, Src: Intel_96:a9:44 (00:19:d2:96:a9:44), Dst: Netgear_1a:ad:fb (c4:3d:c7:1a:ad:fb)
⊞ Internet Protocol Version 4, Src: 192.168.0.3 (192.168.0.3), Dst: 173.194.41.151 (173.194.41.151)
⊞ Transmission Control Protocol, Src Port: 52188 (52188), Dst Port: https (443), Seq: 1, Ack: 1, Len: 37
⊞ Secure Sockets Layer

0000  c4 3d c7 1a ad fb 00 19  d2 96 a9 44 08 00 45 00   .=...... ...D..E.
0010  00 4d 50 cd 40 00 80 06  11 d9 c0 a8 00 03 ad c2   .MP.@... ........
0020  29 97 cb dc 01 bb 44 1d  5d 55 c2 8a 4c 16 50 18   ).....D. ]U..L.P.
0030  00 3d 81 0d 00 00 17 03  01 00 20 03 ef 69 40 9b   .=...... .. ..i@.
0040  cf f6 90 40 ea 24 29 5b  e7 1f 6d 57 ab d9 31 7a   ...@.$)[ ..mW..1z
0050  32 4b e2 69 a1 42 2e c9  df 5d 47                  2K.i.B.. .]G

Microsoft: <live capture in progress> File: C:...   Packets: 1390 Displayed: 1390 Marked: 0
```

Capture of packets in Wireshark

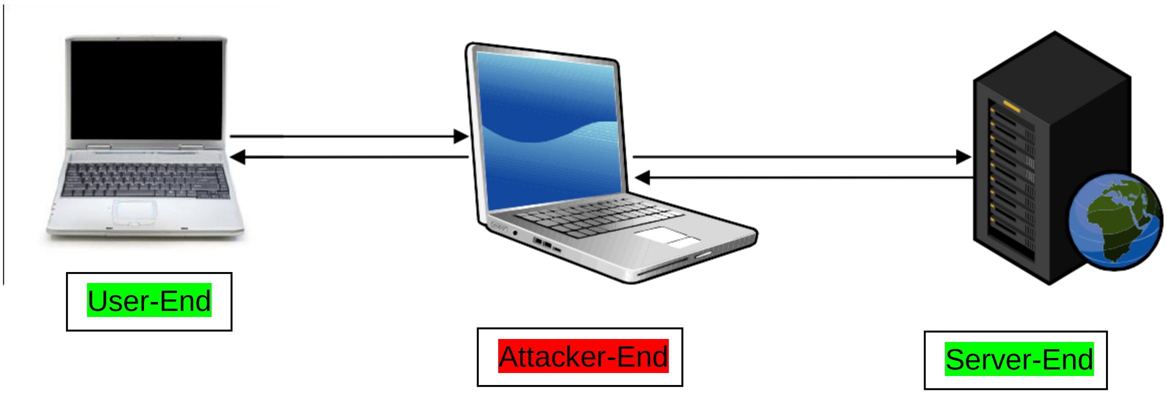


As illustrated in the above figure , it clearly shows how a typical session hijacking attack is conducted between a client and a server by an attacker. The traffic is constantly monitored using the packet capturing tool and then the packets are analyzed to understand which packet contains the session information required to authenticate to server.

3.2 Passive Session Hijacking:

In a passive session the attacker listens to all the data and captures them for future attacks, in most cases to perform any type of a hijacking attack it is important that the attacker starts off with passive mode. The disadvantage in the passive mode attack is that the attacker might not be that efficient in succeeding on the user impersonating to the server, unless the user session is still alive in most cases it will not be, if the user logs off from the server.

Another typical man-in-the-middle attack is session-replay, unlike session hijacking, in a session-replay the attacker captures all the packet and alters the packet information before sending it to the server for the authentication, this a typical man- in-the-middle attack because the attacker is in-fact between the user and server modifying the packet and sending. The following figure (2) shows a typical session replay attack.



Session Replay attack

As shown in the above figure , the attacker interrupts the traffic between the server and client, and modifies before sending it back. This type of attack brings up a lot of suspicion for a network administrator or the user itself. This is lot more time consuming when compared to other types of man-in-the-middle attacks.

3.3 Hybrid Session Hijacking:

In hybrid session hijacking the attacker implements both the modes of attack that is passive and active mode to successfully complete the attacks. In this type of attack, the attacker monitors the pattern of traffic that has been sent over the network and the attacker chooses a session to impersonate. A typical example will be a public unprotected wireless network, where the attacker has access to multiple sessions in progress. All the attacker has to do in this situation is to wait for the right session and hijack the session from the user.

Further the session hijacking attacks can be categorized into 2 different sub types which dependent on the Spoofing techniques used for the attack (Andrew & Daniel, 2006):

- **Blind Spoofing-attack**
- **Non-blind Spoofing-attack**

3.3.1 Blind Spoofing-attack:

Spoofing is a technique of compromising the target machine without attracting any attentions to the attack. When traffic cannot be seen and if the attacker just dumps the traffic between the client and the server, and then by guessing the tcp sequence number tries to authenticate into the server. This makes it a most difficult type of attack to perform, and in most cases the attacker ends up spending a lot of time without any success (Andrew & Daniel, 2006).

3.3.2 Non-Blind Spoofing-attack:

This is the most common type of attack, since in non-blind spoofing the attacker can see the traffic between the client and the server machine. It makes it easy for the attackers to analyze the packets in an active mode and further the attack by impersonating as the user to the server. This becomes difficult in a switched network, as the switches do not broadcast all the packets to all the hosts, rather to a particular host. But with some advanced configuration, if the attacker can compromise the VLAN (Virtual LAN) port then session hijacking is possible in the network

4.Some of the tools used in session hijacking

- ✓ **Hunt**
- ✓ **T-Sight**
- ✓ **Juggernaut**
- ✓ **TTY Watcher**
- ✓ **Hamster and Ferret**
- ✓ **Wireshark**
- ✓ **Ethereal**

5.Detection tools and techniques for Session Hijacking

To protect against session hijacking there are various intrusion detection tools and some advanced techniques. The below list are only a few commonly used tools:

- ✓ **Arp-ON**
- ✓ **ARP-PING**
- ✓ **ANTI-SNIFF**
- ✓ **Cookie Monster**
- ✓ **Wavelet based detection**
- ✓ **Cisco Intrusion Detection System (IDS)**
- ✓ **Sans Intrusion Prevention System (IPS)**

5.1 Arp-ON: This tool is aimed to secure the Address resolution protocol, and avoid any MITM attacks (Darknet, 2000).

5.2 ARP-PING: This is a Linux tool, and allows the user to ping a Media Access Control (MAC) address directly. This can implemented to detect the attacker using a sniffer on the network (Beyond-Security, 1998).

5.3 ANTI-SNIFF: In this tool, the user can detect any sniffer on the network used for packet capturing (Storm, 2011).

5.4 Cookie-Monster: This tool was developed for analyzing the strength of the cookie by archiving and analyzing (Pauli, Engebretson, Ham, & Zautke, 2011).

5.5 Wavelet-based-detection: In this techniques, the author analysis the signal strength using wavelet transform to detect a session hijacking (Long & Sikdar, Wavelet Based Detection of Session Hijacking Attacks in Wireless Networks, 2008).

The proposal in this thesis classifies the detection strategies as the following:

- ✓ **In-Network Strategy**
- ✓ **Out-Network Strategy**

As we have seen every approach in the research field for session hijacking detection mechanism, the authors have either proposed a detection mechanism for a specific network that is within a LAN, or for an outside network. But this paper proposes an approach that will protect targets against session hijacking attacks from both inside the network and also outside the network.

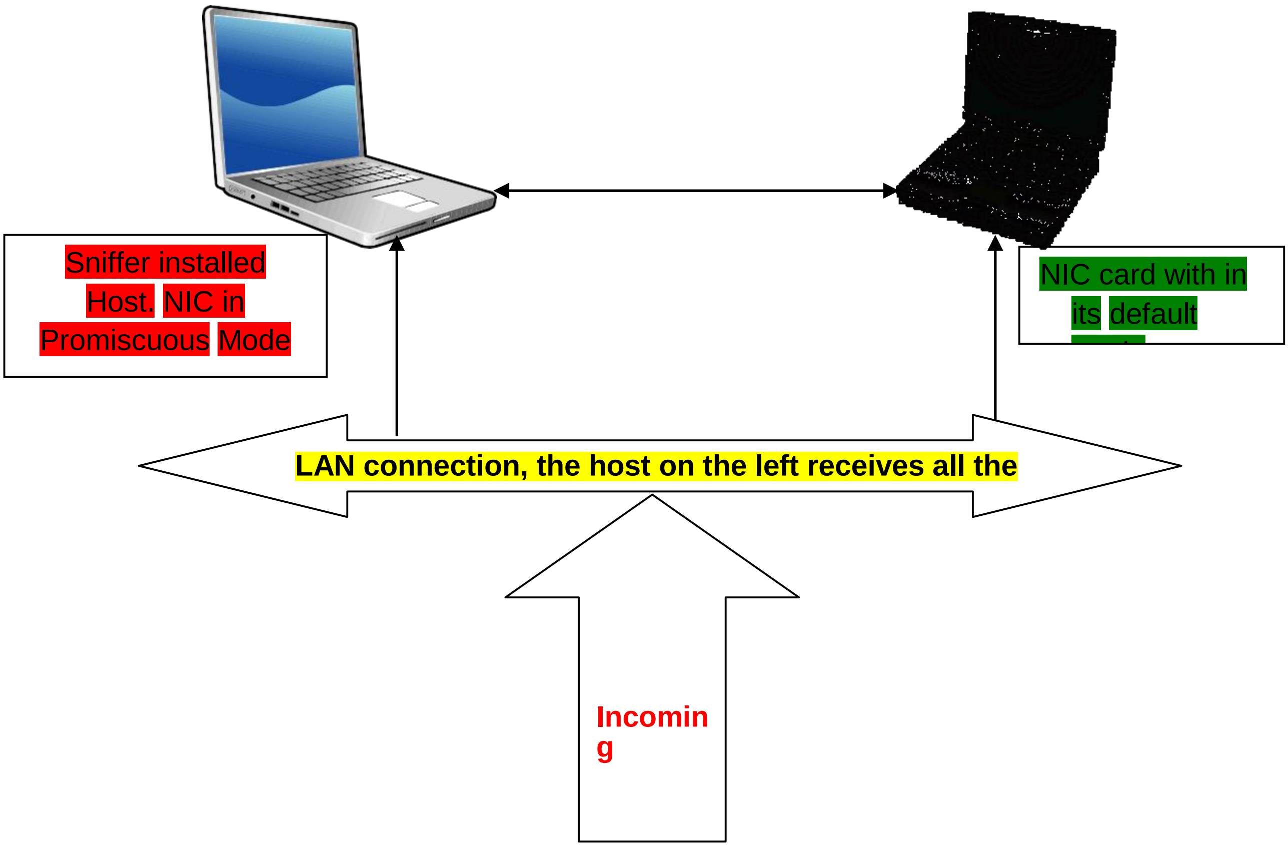
In-Network Strategy:

A network in which the defense is defined for an attack that can occur within a network is known as In-Network Strategy. In the In-Network Strategy the detection of a session hijacking is seen from the client‟s perspective. The proposal defines a mechanism to be adapted in-order to detect any session hijacking attempts from the user-end.

One of the basic criteria for the session hijacking attack, the attacker needs to listen to the traffic be it in active mode or passive mode, and choose a session to hijack. However, in order to listen to the traffic, a sniffer needs to be installed on the attacker machine and this sniffer will listen to all the traffic that traverses through the network. A sniffer tools like wireshark or ethereal needs to set the mode of the Network Interface Card (NIC) to receive all traffic. The basic functionality of Network Interface card is to accept only packets that are designated to it or any broadcast packet to check if the broadcast is for that particular host .To set a NIC card into a mode where it can receive all the traffic even if it is not intended to that host or IP address, such a mode is known as **Promiscuous Mode .**

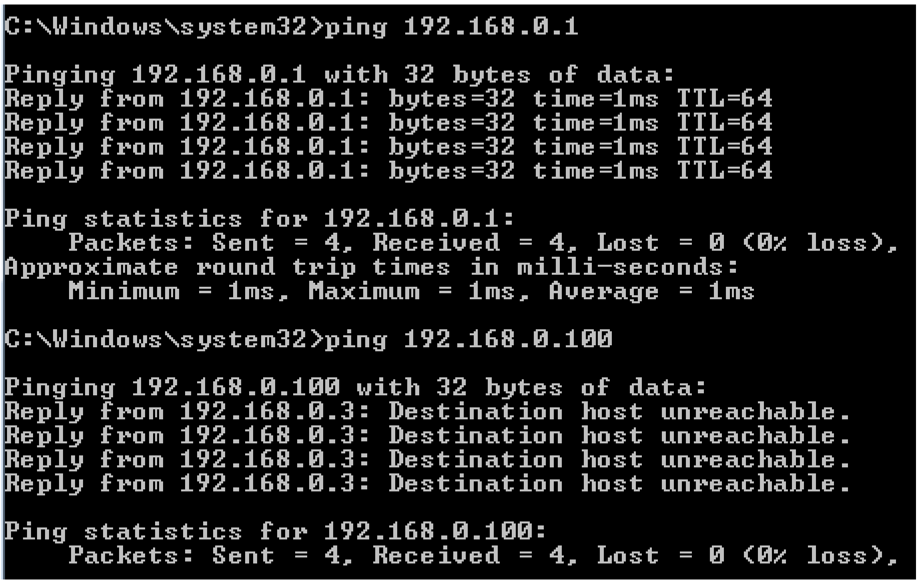
This is independent of what operating system is installed on the host machine, all though there are some restrictions on the windows operating systems.

The strategy proposed in this thesis is to use this characteristic of a sniffer against it to detect the network Interface card in promiscuous mode.


Sniffer installed Host. NIC in Promiscuous Mode
NIC card with in its default
LAN connection, the host on the left receives all the
Incomin g

NIC in Promiscuous Mode accepts all the packets

There are several methods available to detect a sniffer in promiscuous mode; this paper will be using Internet Control Message Protocol (ICMP) to detect a sniffer in the network. ICMP packet is a simple ping packet which echo‟s the response from the host . The echo replies are from the host to which the ping was intended to. If the host is unavailable it will reply a destination not reachable error as shown in the screen capture. Using the ICMP protocol we can send a fake PING packet to the network, a host whose Network Interface Card (NIC) is set to promiscuous mode is designated to receive all the packets from that network. This is where we will try and manipulate the Sniffer making it believe that it is a legitimate packet and forcing it to send a response to the ping packet.



```
C:\Windows\system32>ping 192.168.0.1

Pinging 192.168.0.1 with 32 bytes of data:
Reply from 192.168.0.1: bytes=32 time=1ms TTL=64
Reply from 192.168.0.1: bytes=32 time=1ms TTL=64
Reply from 192.168.0.1: bytes=32 time=1ms TTL=64
Reply from 192.168.0.1: bytes=32 time=1ms TTL=64

Ping statistics for 192.168.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 1ms, Average = 1ms

C:\Windows\system32>ping 192.168.0.100

Pinging 192.168.0.100 with 32 bytes of data:
Reply from 192.168.0.3: Destination host unreachable.
Reply from 192.168.0.3: Destination host unreachable.
Reply from 192.168.0.3: Destination host unreachable.
Reply from 192.168.0.3: Destination host unreachable.

Ping statistics for 192.168.0.100:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

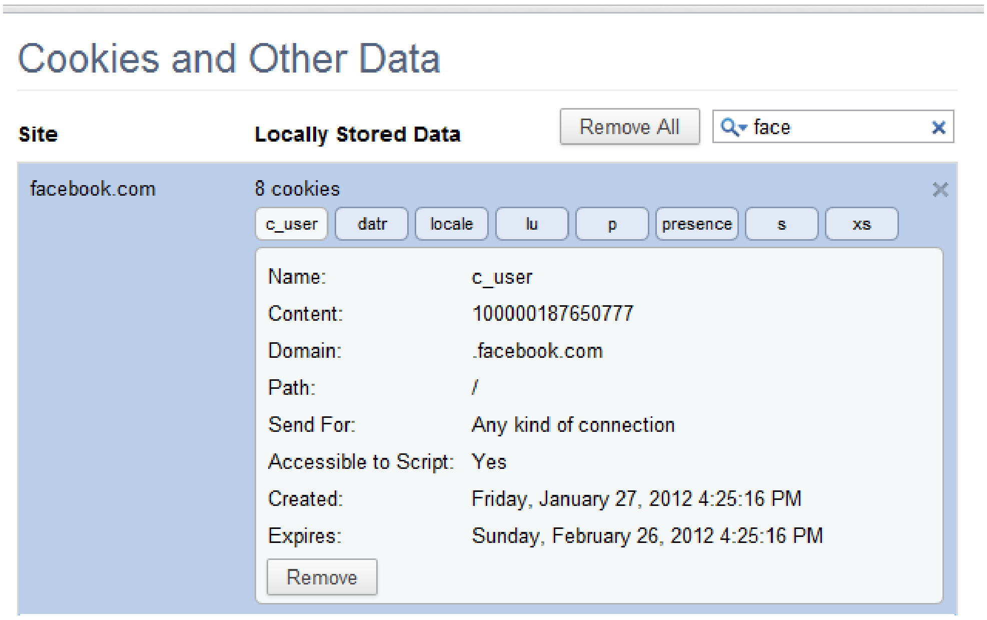
Ping results for successful reply and also unsuccessful response.

Out-Network Strategy:

A network in which the defense is defined for an attack that occurs from outside the network is known as Out-Network Strategy. In the OUT-Network Strategy the detection of a session hijacking is seen from the Server‟s perspective. The proposal defines a mechanism to be adapted in-order to detect any session hijacking attempts from the server-end.

The server can play a crucial role in defining and detecting an attack, since the server only reads the session information in a cookie as it comes to it and processes it without having to notice from where it is come. If there are username and password included in the cookie, the server processes them to a database for instance and cross verifies if the credentials are correct and then serves the request content with a session ID, for that particular session. Now it‟s the attackers turn, who is watching all this process happen, now the attacker swoops in and steal the session information after the

initial authentication process has been completed and uses this session information to successfully receive the contents from the server. Typical cookie information is as shown in the following screenshot (3) this screenshot was taken from a Google chrome browser after authenticating to the Facebook server.



Cookies and Other Data

Site: facebook.com

Locally Stored Data: 8 cookies

Name: c_user
Content: 100000187650777
Domain: .facebook.com
Path: /
Send For: Any kind of connection
Accessible to Script: Yes
Created: Friday, January 27, 2012 4:25:16 PM
Expires: Sunday, February 26, 2012 4:25:16 PM

Cookie information from a local browser

As we can see there are more than one cookie created for each session, but the session ID will remain the same throughout that particular session. Now for each host outside the network, we will have a unique IP address and port no‟s used in-order to interact with the server, and that is how a server identifies as to where the request is coming from and where the request needs to be send.

Using the uniqueness of the IP-address on the internet we can differentiate the session requests from various hosts, now the fundamental question will arise as to how to save and use this IP address against each request. For every request the user makes to the server, we will assume that the initial request made to the server at a specific point of time, from a specific host and identify this as unique. Now we use the filter technique to separate the genuine requests from false requests or hijacked requests. The details of this implementation are explained in the test-bed section.

User System

The following are specification of the user system used for the testing:

- Operating System: Windows 7
- IBM ThinkPad T60 laptop
- Processor: Intel Dual Core – 1.6GHz
- Memory: 2GB RAM
- Hard disk: 160 GB HDD
- A local browser

Server System

For the experiment of this implementation we used the following specifications:

- Server Installed: Apache Tomcat Version 7.0.5
- Server has been Installed on a windows 7 operating system
- The server here has a localhost identity
- And it is in the same machine as the user.

Attacker System

Attacker system is the specification:

- Back-Track Linux operating system
- Ferret and Hamster tool to perform the attack
- Intel Core i3 processor – 2.0Ghz
- Memory: 3GB
- Hard Disk: 500 GB

The above specifications are some of the main design requirements used in order to conduct the testing of this approach. There were constant updating and alternate patches were required to be used at some stage during the testing.

6. What Can Attackers Do After Successful Session Hijacking?

- The attacker can perform any action that the user was carrying out with his credentials.
- The hacker can gain access to multiple web applications, from financial systems and customer records to line-of-business systems potentially containing valuable intellectual property.
- The attacker can use session hijacking cookies for identifying authenticated users in single sign-on systems (SSO).
- Here are a few examples:
  - Attackers can log into bank accounts for transferring money
  - Hackers can use the access for online shopping
  - Hackers can get access to sensitive data and sell it on the dark web
  - Hackers can demand a ransom from the user in exchange for the data

7. Prevention of Session hijacking

- Session hijacking can be protected by taking preventive measures on the client side.
- Software Updating, End Point Security will be a key from a user side.
- Having Biometric authentication for every user session can prevent attacks.
- End to End encryption can be done between the user browser and web server using secure HTTP or SSL.
- We can have the session value stored in the session cookie.
- We can have an automatic log off after the session ends.
- We can use session ID monitors.
- VPN use can prevent unauthorized access.
- Web server generating long random session cookies can prevent attacks.
- Usage of Session ID monitors enhances security.
- Deleting the session cookie from the user server and computer enhances security.
- Having different HTTP header order for different sessions is a good precaution.

In order to protect from being hijacked while in a session, you need to strengthen the mechanisms in web applications. This can be done through communication and session management. Here are a few ways you can reduce the risk of session hijacking:

7.1 HTTPS: The use of HTTPS ensures that there is SSL/TLS encryption throughout the session traffic. Attackers will be unable to intercept the plaintext session ID, even if the victim’s traffic was monitored. It is advised to use HSTS (HTTP Strict Transport Security) to guarantee complete encryption.

7.2 HTTP Only: Setting up an HTTP Only attribute prevents access to the stored cookies from the client-side scripts. This can prevent attackers from deploying XSS attacks that rely on injecting Java Scripts in the browser.

7.3 System Updates: Install reputable antivirus software which can easily detect viruses and protect you from any type of malware (including the malware attackers use to perform session hijacking). Keep your systems up to date by setting up automatic updates on all your devices.

7.4 Session Management: In order to offer sufficient security, website operators can incorporate web frameworks, instead of inventing their own session management systems.

7.5 Session Key: It is advised to regenerate session keys after their initial authentication. This renders the session ID extracted by attackers useless as the ID changes immediately after authentication.

7.6 Identity Verification: Perform additional identity verification from the user beyond the session key. This includes checking the user's usual IP address or application usage patterns.

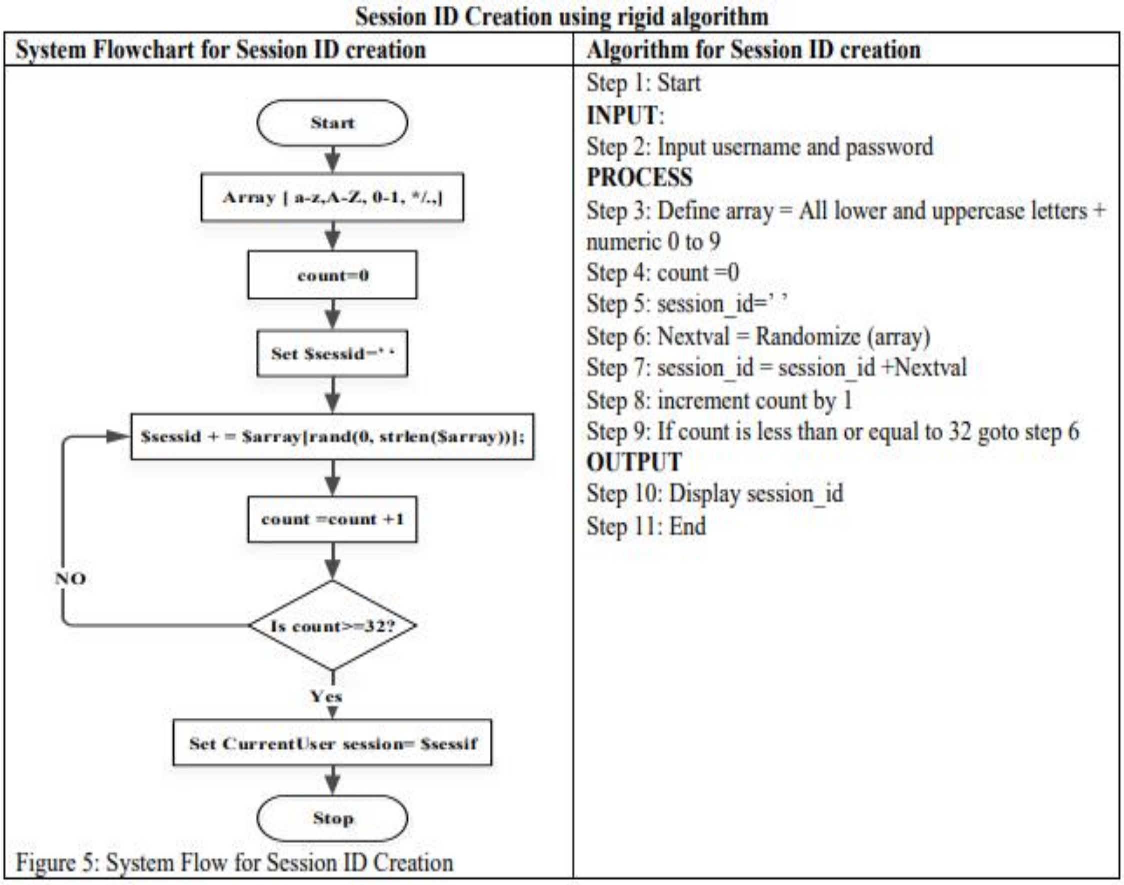
7.7 Public Hotspot: Avoid using public WiFi to protect the integrity of your sessions and opt for secure wireless networks.

7.8 VPN: Use a Virtual Private Network (VPN) to stay safe from session hijackers. A VPN masks your IP and keeps your session protected by creating a “private tunnel” through which all your online activities will be encrypted.

7.9 Phishing Scam: Avoiding falling for phishing attacks. Only click on links in an email that you have verified to have been sent from a legitimate sender.

7.10 Session ID creation using rigid algorithm

Fragile/Short session ID's can be expose to attacks. The application of cryptographic algorithm can enable us to detect the attack. The attacker can study the session ID generation to drawn knowledge in creating a new session. Hence, to avoid this risk, an algorithm to generate long random alphanumeric character is used as a session key.

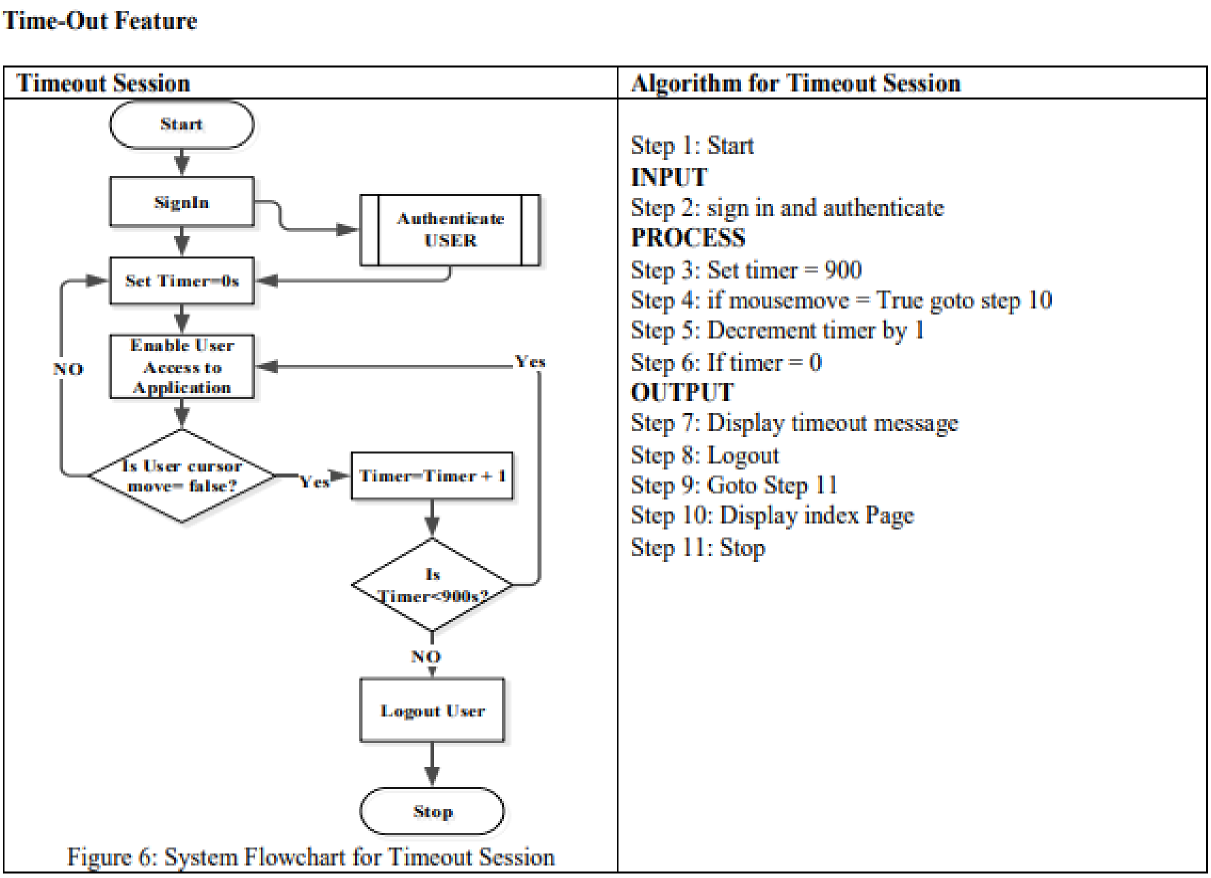

Start
Array [a-z,A-Z, 0-1, */.,]
count=0
Set $sessid=' '
$sessid + = $array[rand(0, strlen($array))];
count =count +1
NO
Is count>=32?
Yes
Set CurrentUser session= $sessif
Stop

Session ID Creation using rigid algorithm

System Flowchart for Session ID creation	Algorithm for Session ID creation
Figure 5: System Flow for Session ID Creation	Step 1: Start **INPUT**: Step 2: Input username and password **PROCESS** Step 3: Define array = All lower and uppercase letters + numeric 0 to 9 Step 4: count =0 Step 5: session_id=' ' Step 6: Nextval = Randomize (array) Step 7: session_id = session_id +Nextval Step 8: increment count by 1 Step 9: If count is less than or equal to 32 goto step 6 **OUTPUT** Step 10: Display session_id Step 11: End

7.11 Timing out Sessions

If the system is left idle and user didn’t perform any operation on it and did not log out. Attacker can steal the session ID and thereby hijack the session. It is therefore necessary for user to timeout after a constant period of time if idleness to prevent from attack. In the system developed the timeout session is set to 15 minutes as a stronger control, leaving not enough time for the attacker to penetrate.

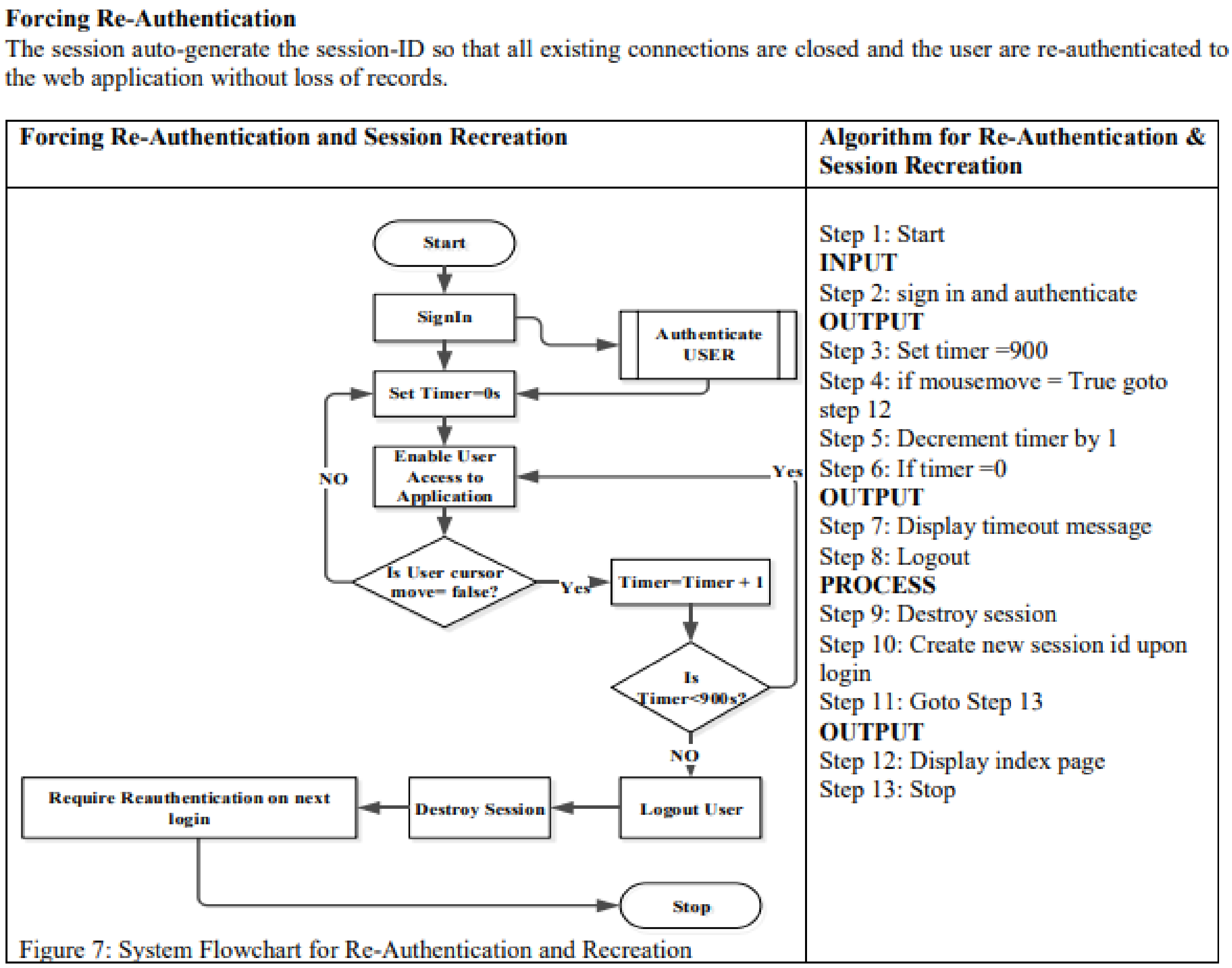


Time-Out Feature

Timeout Session	Algorithm for Timeout Session
Figure 6: System Flowchart for Timeout Session	Step 1: Start **INPUT** Step 2: sign in and authenticate **PROCESS** Step 3: Set timer = 900 Step 4: if mousemove = True goto step 10 Step 5: Decrement timer by 1 Step 6: If timer = 0 **OUTPUT** Step 7: Display timeout message Step 8: Logout Step 9: Goto Step 11 Step 10: Display index Page Step 11: Stop

7.12 Forcing Re-authentication

This module enable user to access the system after a constant period of time to relogin. Here, the session ID is recreated and connection established. The previous session ID becomes outdated and it is



Forcing Re-Authentication

The session auto-generate the session-ID so that all existing connections are closed and the user are re-authenticated to the web application without loss of records.

Forcing Re-Authentication and Session Recreation	**Algorithm for Re-Authentication & Session Recreation**
Figure 7: System Flowchart for Re-Authentication and Recreation	Step 1: Start **INPUT** Step 2: sign in and authenticate **OUTPUT** Step 3: Set timer =900 Step 4: if mousemove = True goto step 12 Step 5: Decrement timer by 1 Step 6: If timer =0 **OUTPUT** Step 7: Display timeout message Step 8: Logout **PROCESS** Step 9: Destroy session Step 10: Create new session id upon login Step 11: Goto Step 13 **OUTPUT** Step 12: Display index page Step 13: Stop

8.Mitigation of Session Hijacking

There are various types of methods it depends on the types of attack that we are trying to mitigate. Some of the common methods are:

- ✓ **Intrusion detection Mechanisms**
- ✓ **Firewall**
- ✓ **User Awareness**
- ✓ **Security Testing Methodology**
- ✓ **Virtual Private Networks**
- ✓ **Physical Security**

8.1 Intrusion Detection Mechanisms

An intrusion detection system protects an infrastructure at two different level like network level and host machine level. Developers should take careful consideration is developing customized IDS‟s for their organizations. For network level, IDS will have to be implemented between the routers and servers, also depending on the level of traffic received. For a typical host system, the IDS can be implemented between the user machines and local network (Bottino, 2006).

8.2 Firewall:

Firewalls are ideal for analyzing the incoming and outgoing traffics of a network. Firewall can be installed both within the network and outside the network. In a firewall a rule based definition can be given to what type of incoming traffic are allowed or denied, similarly what type of outgoing traffics are allowed/denied in that particular network, this a filter based approach and can every effective if correctly implemente

8.3 User Awareness:

A system can have a very sophisticated system security system, but if the users are not aware of their responsibility regarding the do‟s and don‟ts on the network, it will become a very serious security threat. The user‟s should be made aware about the restricted the devices also the restricted information to be given out.

8.4 Security Testing Methodology:

To achieve maximum security a good testing methodology should be followed in order to evaluate the infrastructure security level and any possible loopholes. For big organization it‟s important to gets a well security testing methods regularly to keep their system up-to-date like vulnerability assessment and Penetration testing.

A vulnerability testing is conducted to assess the systems for any possible security loopholes, while penetration testing is conducted to exploit the system to check for any possible vulnerability.

8.5 Virtual Private Network:

A virtual private network is a private network that is implemented over the public channel to connect to a remote host. By enabling this service we can protect any resources that will access from outside the infrastructure via internet in an encrypted manner using various cryptography technologies.

8.6 Physical Security:

A physical security is the security given to the location where the data center, sever room or any information storage area, office infrastructure is situated. It is important secure the physical location, since any damage done on a physical location will never be recoverable and it‟s lost forever. Security alarms, ID cards for employees, biometric entry, etc. are some of the technologies that can be used.