Technique

ARP (My Note)

Certainly! This Python code uses the Scapy library to perform an ARP (Address Resolution Protocol) scan on a local network. The purpose of an ARP scan is to discover live hosts within a specified IP range. Let's break down the code step by step:

```
from scapy.all import *

# Set the network interface, IP range, and broadcast MAC addr
ess
interface = "eth0"
 ''' please check the net interface before putting this! for
my case it was ens5'''

ip_range = "10.10.X.X/24"
'''you might have to change the ip address or you can use the
'''
broadcastMac = "ff:ff:ff:ff:ff:ff"

# Create an ARP request packet
packet = Ether(dst=broadcastMac) / ARP(pdst=ip_range)

# Send the packet and wait for responses
ans, unans = srp(packet, timeout=2, iface=interface, inter=0.
1)

# Print the results
for send, receive in ans:
    print(receive.sprintf(r"%Ether.src% - %ARP.psrc%"))
```

Now let's break down each part of the code:

1. **Import Scapy:**

- `from scapy.all import *`

  This line imports all symbols from the Scapy library, which is a powerful packet manipulation and network scanning tool in Python.

- **Configuration:**

- ```
  interface = "eth0"
  ip_range = "10.10.X.X/24"
  broadcastMac = "ff:ff:ff:ff:ff:ff"
  ```

  Define the network interface (`interface`), the IP range to scan (`ip_range`), and the broadcast MAC address (`broadcastMac`).

- **Create ARP Packet:**

```
packet = Ether(dst=broadcastMac) / ARP(pdst=ip_range)
```

This line creates an ARP request packet using Scapy. It sends a broadcast ARP request to all IP addresses in the specified range.
```

This thing needs to be highlighted.

Certainly! Let's break down the line of code that creates an ARP request packet using Scapy:

```
packet = Ether(dst=broadcastMac) / ARP(pdst=ip_rang
e)
```

1. `Ether` **Layer:**
    - `Ether` is a Scapy class representing the Ethernet layer of the network stack.
    - `Ether(dst=broadcastMac)` creates an Ethernet frame with a specified destination MAC address (`dst`).
    - In this case, `broadcastMac` is a variable holding the broadcast MAC address (`"ff:ff:ff:ff:ff:ff"`), which means the frame is intended for all devices on the local network.
2. `ARP` **Layer:**
    - `ARP` is a Scapy class representing the Address Resolution Protocol layer.
    - `/ ARP(pdst=ip_range)` appends an ARP layer to the Ethernet frame, specifying the ARP packet's target IP address (`pdst`).
    - `ip_range` is a variable that holds the IP address range to be scanned. It's often in the form of `X.X.X.X/Y`, where `X.X.X.X` is the IP address, and `Y` is the subnet mask.

Putting it all together, this line of code creates an ARP request packet encapsulated in an Ethernet frame. The ARP request is broadcast to all devices on the local network, and the target IP address specified in the ARP packet is set to the IP range specified in `ip_range`.

This type of ARP request is often used in network scanning to discover live hosts on a local network. The broadcast MAC address ensures that

the request is sent to all devices, and the target IP address range defines the scope of the scan.

- **Send and Receive Packets:**
- `ans, unans = srp(packet, timeout=2, iface=interface, inter=0.1)`

  The `srp` function sends the ARP request packet and waits for responses. It returns two lists: `ans` contains the packets that received a response, and `unans` contains packets without a response.
- **Print Results:**

1. `for send, receive in ans: print(receive.sprintf(r"%Ether.src% - %ARP.psrc%"))`

   Iterate through the received packets and print the source MAC address (`%Ether.src%`) and the corresponding IP address (`%ARP.psrc%`). This information represents live hosts on the network.

Note: Replace "10.10.X.X/24" with the actual IP range you want to scan. Also, ensure you have the necessary permissions to send and receive packets on the specified network interface.

```
┌──(root💀kali)-[/home/alper/Desktop]
└─# python3 arp_scan.py
Begin emission:
Finished sending 256 packets.
.******.......................
Received 32 packets, got 6 answers, remaining 250 packets
a8:5e:45:72:0a:38 - 192.168.1.1
c0:e4:34:bd:af:c9 - 192.168.1.3
dc:41:a9:62:ca:50 - 192.168.1.5
2c:6f:c9:18:79:0f - 192.168.1.2
98:2c:bc:4b:7e:36 - 192.168.1.8
de:e5:e4:19:3b:c6 - 192.168.1.9
```

If you are using the AttackBox, you will need to install Scapy first. This can easily be done using the "`apt install python3-scapy`" command.

```
root@ip-10-10-143-223:~# apt install python3-scapy
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libjs-sphinxdoc libjs-underscore
Suggested packages:
  python3-matplotlib ipython3
The following NEW packages will be installed
  libjs-sphinxdoc libjs-underscore python3-scapy
```