

EX NO: 7	SCAPY
DATE: 22/09/2023	

Aim: To install and see the working of Scapy.

Procedure:

Installation of scapy module:

As scapy module is not included in Python3 library by default, we have to add it into our Python library using pip. Execute this command in your Linux terminal to get the scapy module for Python3.

```
pip3 install scapy-python3
```

Some important functions for creating Network scanner

ARP(): This function defined in scapy module which allows us to create ARP packets (request or response). By default, if we are calling it, it will create an ARP request packet for us.

```
import scapy.all as scapy request
```

```
-scapy.ARP()
```

Summary(): This method provide us the status of the packet that we have created. It does not provide the detailed information about the packet, it just gives us the basic idea like what is the type of packet, what is the destination of the packet etc. For example if we want to create an ARP packet using ARPO method which is present in the scapy module and want to see the summary of the packet then we can do this by creating the object of ARP class

```
import scapy.all as scapy
```

```
)
```

```
request=scapy.ARP
```

```
(print(request.summary()))
```

Now we have created a request packet of ARP. Here the output of the program will be like this

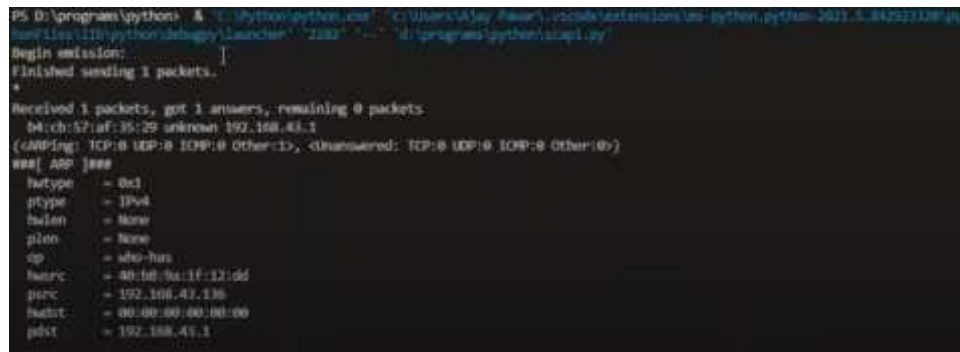
```
root@kali:~/Desktop# python3 networkscanner.py ARP
```

```
who has 0.0.0.0 says 192.168.215.144
```

```
root@kali:~/Desktop#
```

Show() method: This method is very similar to summary() method. It gives more detailed information about the packet. The usage of this function is also much similar to as summary() method.

```
import scapy.all as scapy
request= scapy.ARP()
print(request.show())
```



```
PS D:\program\python > & "C:\Python\python.exe" "C:\Users\Aley\OneDrive\code\extensions\ms-python\python-3621.5.84223128\pyt
her\lib\site-packages\scapy\launcher.py" "2122" "..." "d:\program\python\scapy.py"
Begin session:
Finished sending 1 packets.
Received 1 packets, got 1 answers, remaining 0 packets.
[eth:52:af:35:29: unknown: 192.168.43.1]
(<ARPing: TCP:0 UDP:0 ICMP:0 Other:1>, <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
use[ ARP ]use
  htype      = 0x1
  ptype      = 1904
  hlen       = None
  plen       = None
  op         = who-has
  hwsrc      = 48:6d:8a:1f:12:dd
  psrc       = 192.168.43.138
  hdst       = 00:00:00:00:00:00
  pdst       = 192.168.43.1
```

Is() function: This method is present in the scapy class. By using this method, we can see what are the fields that we can set for a specific packet.

In our example we will create an ARP packet and the with the help of ls() function, we will see what are the available fields for this packet.

```
import scapy.all as scapy
request =scapy.ARPQ
print(scapy.ls(scapy.ARPQ))
```

1. Create an ARP packet using ARP() method.

2. Set the network range using variable.
3. Create an Ethernet packet using Ether() method.
4. Set the destination to broadcast using variable hwdst.
5. Combine ARP request packet and Ethernet frame using /".
6. Send this to your network and capture the response from different devices.
7. Print the IP and MAC address from the response packets.

Below is the Python implementation

```
import scapy.all as scapy
request = scapy.ARP(
request.pdst = 'x'
broadcast=scapy.Ether()
broadcast.dst = 'ff:ff:ff:ff:ff:ff'
request_broadcast = broadcast / request
clients=scapy.srp(request_broadcast, timeout=1)[0]
for element in clients:
print(element[1].psrc + "
"+element[1].hwsrc)
```