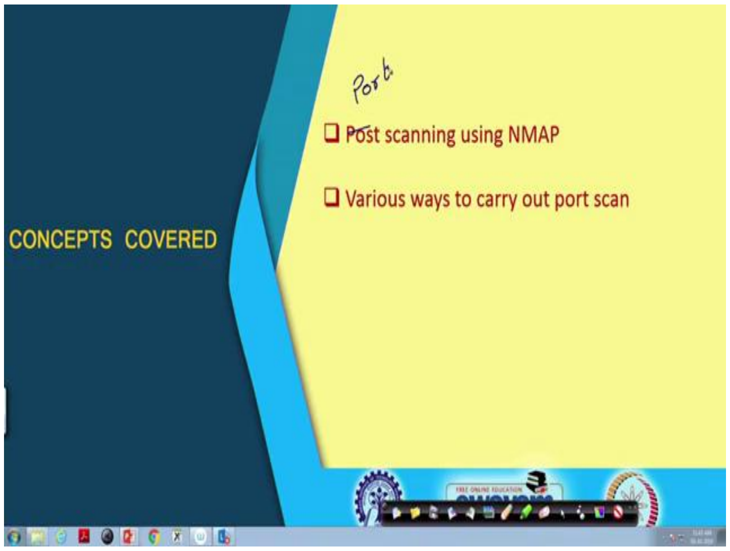
Ethical Hacking

Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 57
The NMAP Tool : A Relook (Part - II)

We continue with our discussion on NMAP. If you recall, in the last lecture we talked about specifically the host discovery kind of commands; that is there.

(Refer Slide Time: 00:29)

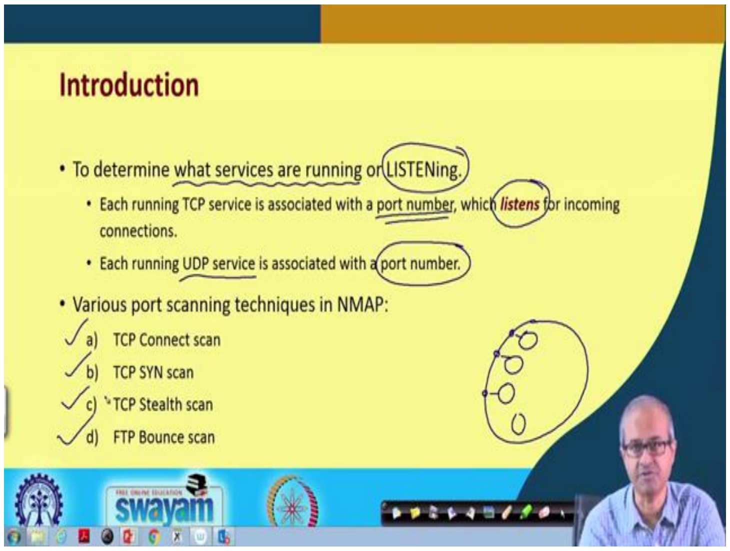


CONCEPTS COVERED

- ~~Post~~ Port scanning using NMAP
- Various ways to carry out port scan

But, here in the part II of the lecture, we shall be mainly talking about post scanning; sorry this will be port, port, port scanning using NMAP and so this is actually port. It is the typographic error and various ways in which we can carry out this port scanning ok.

(Refer Slide Time: 00:50)



Introduction

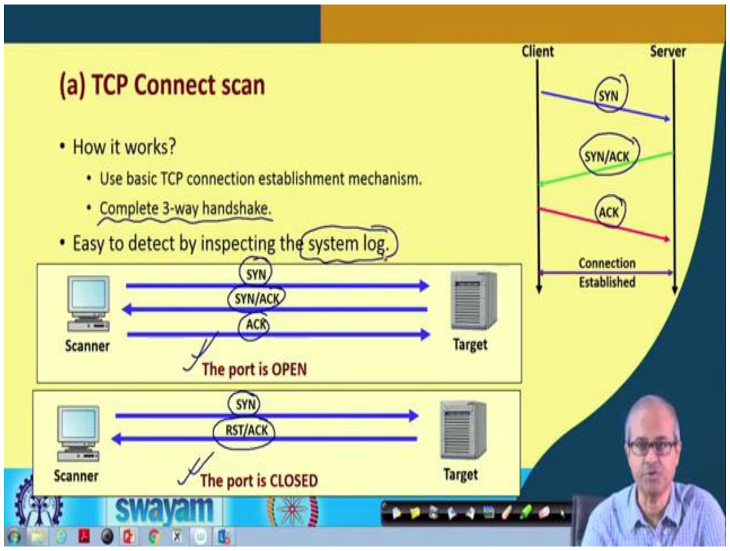
- To determine what services are running or LISTENing.
  - Each running TCP service is associated with a port number, which listens for incoming connections.
  - Each running UDP service is associated with a port number.
- Various port scanning techniques in NMAP:
  - a) TCP Connect scan
  - b) TCP SYN scan
  - c) TCP Stealth scan
  - d) FTP Bounce scan

So, let us see; first thing I mentioned during the last lecture, what do you mean by a port is open or not? Suppose, when you have a server machines, suppose these are server; there can be several services which are running on this machines. And each of the services will be listening to a particular port number; like I said, service like telnet usually listens to port number 23; mail SMTP listens to port number 25 and so on.

So, we say that the server or service is listening on that port number; whenever there is an incoming request on that port number, the request is forwarded to that server. So, here we are basically trying to find out what services are running, which means they are listening on a particular port number. As I had said, each running TCP service is associated with a specific port number where the server or service listens for incoming connections as I had said. In contrast for UDP services, only the port number is associated with; it does not listen on that port.

So, the way UDP and TCP servers are implemented are slightly different. TCP servers listen on a port; UDP servers do not listen. So, it is like connection less; we call it ok. Now, there are several port scanning options which are available in NMAP; some of these we shall be talking about, TCP Connect scan, TCP SYN scan, TCP Stealth and FTP Bounce scan ok. Let us see about this.

(Refer Slide Time: 02:45)



(a) TCP Connect scan

- How it works?
  - Use basic TCP connection establishment mechanism.
  - Complete 3-way handshake.
- Easy to detect by inspecting the system log.

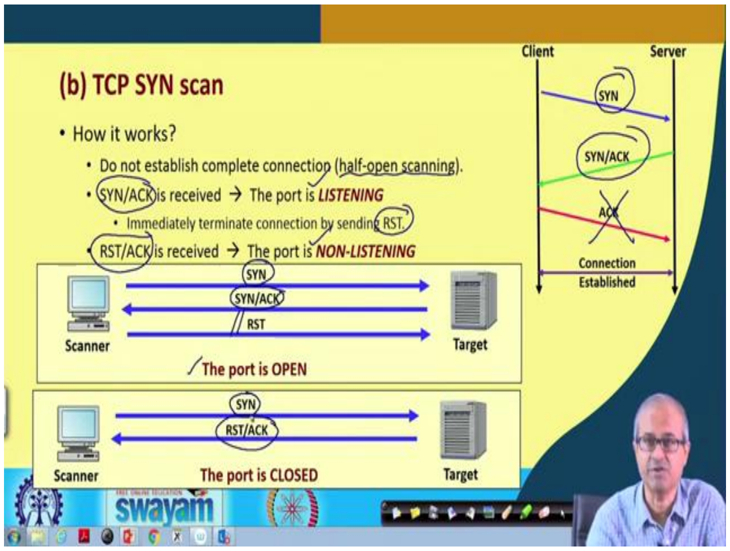
First we talk about TCP Connect scan; but before we talk about it, let us recall; on the right hand side, we have shown the connection establishment phase in TCP. When the client wants to establish a connection to the server, there are 3 packets which flow back and forth. The first packet goes with the SYN flag set; we call it a SYN packet. A packet comes back from the server with both SYN and SCK flag set; we call it SYN/ACK packet. And finally, the client sends back a ACK packet with ACK flag set and then we say that the connection has been established.

So, there are, this is a 3 way handshake protocol. In TCP Connect scan, we basically utilize this mechanism of connection establishment ok. So, the attacker or the person who is mounting this kind of host discovery or port scanning kind of a thing, tries to complete the 3 way handshake on particular port numbers. And, if it finds that this kind of 3 way handshake is successful, this will mean that the particular port number is presently open; there is a server which is presently listening on that port number ok.

So, pictorially I am trying to show it here; that the scanner is first sending a SYN packet on a particular port number. The target if it, if the service is running on the particular port, it will be sending back a SYN/ACK and scanner will finally send back an ACK. So, you conclude that the port is open. But, if you send a SYN packet to a particular port and if the particular port is not having an associated server running, service running, the machine typically sends and reset acknowledgement pack packet back with a reset flag set, indicating that it wants to terminate the connection; the server is not there. So, if such a packet comes back you conclude that the port is closed.

Now, the point to note is that, this kind of a mechanism can detect a port is open or not, but the system administrator can easily detect this by looking at the system log; because all TCP connections are logged in the system log and this is a valid kind of a system connection, TCP connection, you are trying to establish. So, this will also go into the log, fine.

(Refer Slide Time: 05:36)



(b) TCP SYN scan

- How it works?
  - Do not establish complete connection (half-open scanning).
  - SYN/ACK is received → The port is *LISTENING*
    - Immediately terminate connection by sending RST.
  - RST/ACK is received → The port is *NON-LISTENING*

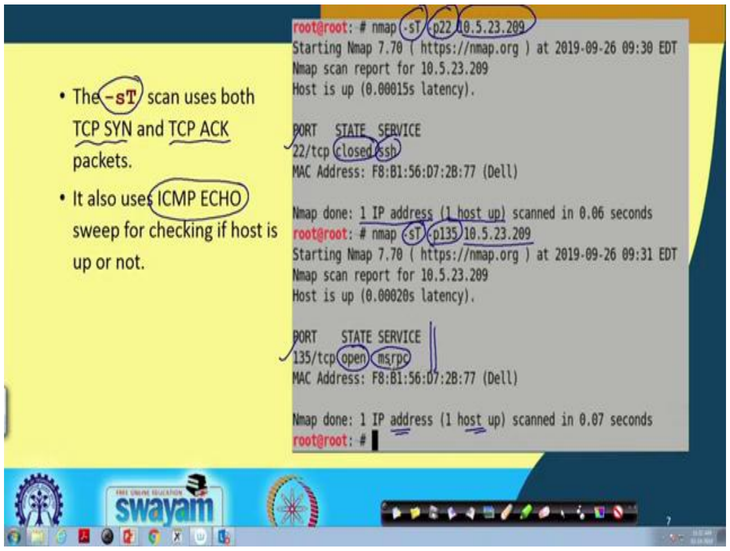
There is a another kind of, this kind of you can say, port discovery mechanism, called TCP SYN scan. Here the mechanism is slightly different; well here also you look; you are utilizing the 3 way handshake; but we are sending a SYN; we are getting back SYN/ACK; but the third ACK we are not sending; that means, we are not completing the connection. If the connection is not completed, then the information will not go into the log. So, you can also escape detection in that way. So, you do not establish the complete connections; sometimes it is called half-open scanning.

So, if you send a SYN packet, if you see that SYN/ACK is received then you conclude that the port is listening. You do not complete the connection; instead of sending an ACK, the scanner immediately terminates connection by sending a reset packet; if a reset packet is sent the connection is terminated immediately. But, if this SYN/ACK or if the server is not up on that port then instead of SYN/ACK, a RST/ACK will be received.

Then you conclude that the port is not listening. Pictorially you show it like this; in the first case when the port is open, you send SYN, you get back SYN/ACK; you immediately terminate connection by sending reset. You do not allow the connection to complete; because your object is not to complete the connection right; Just to see whether the port is open or not.

If the port is closed, then if you send SYN, you will be getting back reset acknowledgement; that is how this works.

(Refer Slide Time: 07:41)



- The -sT scan uses both TCP SYN and TCP ACK packets.
- It also uses ICMP ECHO sweep for checking if host is up or not.

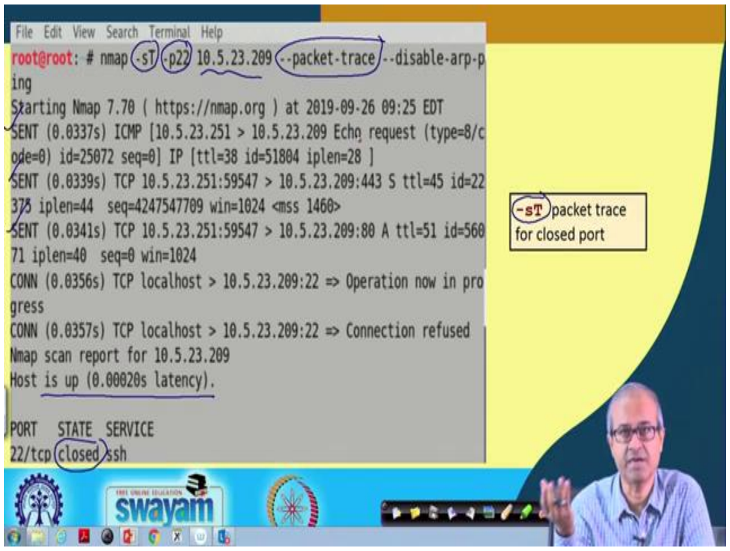
```
root@root: # nmap -sT -p22 10.5.23.209
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-26 09:30 EDT
Nmap scan report for 10.5.23.209
Host is up (0.00015s latency).

PORT   STATE  SERVICE
22/tcp closed ssh
MAC Address: F8:B1:56:D7:2B:77 (Dell)

Nmap done: 1 IP address (1 host up) scanned in 0.06 seconds
root@root: # nmap -sT -p135 10.5.23.209
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-26 09:31 EDT
Nmap scan report for 10.5.23.209
Host is up (0.00020s latency).

PORT    STATE SERVICE
135/tcp open  msrpc
MAC Address: F8:B1:56:D7:2B:77 (Dell)

Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds
root@root: #
```

So, let us look at some examples; both these scans I am showing together. There is an option in NMAP, **-sT** which uses both TCP SYN and TCP ACK request packets. It also uses ICMP; that means, it uses multiple ways of discovering a host and also looking for the port numbers. So, here I am showing a couple of examples; here we are using **-T**, on **sT** with port number 22, on a particular host.

So, you see after scanning, the conclusion is, this port number 22 on TCP is currently closed. Because we are not getting back a response. So, NMAP, 1 IP address, 1 host up, the host was up; but this particular service was closed. Let us take another example, similarly **sT** on port number 135 on some other IP address; here it says that this service is presently open. And also the service number you can see in both cases; what this port number corresponds to ok.

So, here also it says that 1 IP address is scanned. The host was up and also this shows that this service was also up; the port number is open ok.

(Refer Slide Time: 09:18)

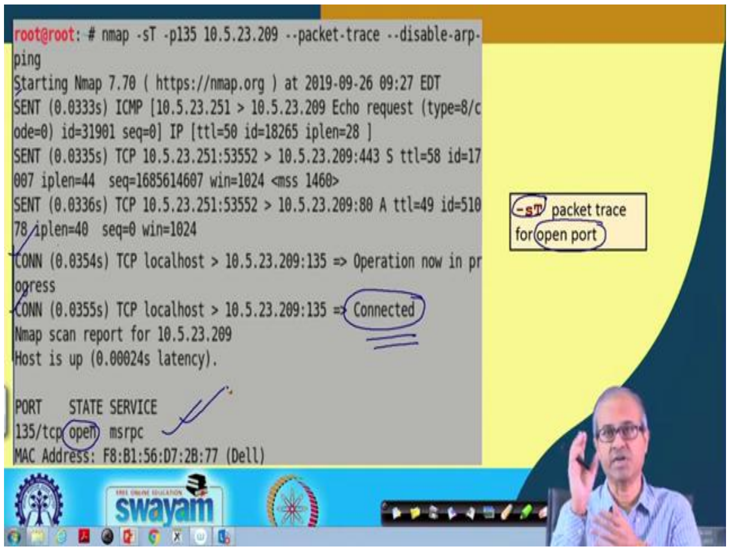


```
File  Edit  View  Search  Terminal  Help
root@root: # nmap -sT -p22 10.5.23.209 --packet-trace --disable-arp-p
ing
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-26 09:25 EDT
SENT (0.0337s) ICMP [10.5.23.251 > 10.5.23.209 Echo request (type=8/c
ode=0) id=25072 seq=0] IP [ttl=38 id=51804 iplen=28 ]
SENT (0.0339s) TCP 10.5.23.251:59547 > 10.5.23.209:443 S ttl=45 id=22
375 iplen=44  seq=4247547709 win=1024 <mss 1460>
SENT (0.0341s) TCP 10.5.23.251:59547 > 10.5.23.209:80 A ttl=51 id=560
71 iplen=40  seq=0 win=1024
CONN (0.0356s) TCP localhost > 10.5.23.209:22 => Operation now in pro
gress
CONN (0.0357s) TCP localhost > 10.5.23.209:22 => Connection refused
Nmap scan report for 10.5.23.209
Host is up (0.00020s latency).

PORT    STATE   SERVICE
22/tcp  closed  ssh
```

Let us take another example. Here we are giving **-sT** with the **packet-trace** option; so that you can get the detail. Here also you are scanning port number 22 on a particular host. So, see here you can see all the packet trace; what packet was sent? Ok, so, all the details, the type of the packet, id, sequence number, for the IP packet the time to live, id, IP length, everything. The final conclusion is host is up, but this service is closed. So, you sent see, 3 TCP packets were sent and based on that your conclusion is, the packet is or the particular port is closed.

So, if you analyze the packets, you will know that why this conclusion is there; you look at the flags; whatever is coming, you can conclude that the other service, not responding fine.

(Refer Slide Time: 10:33)



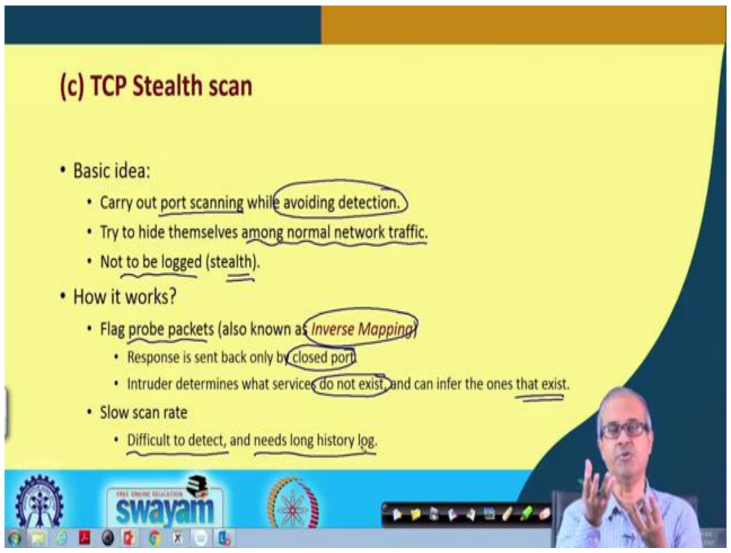
```
root@root: # nmap -sT -p135 10.5.23.209 --packet-trace --disable-arp-
ping
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-26 09:27 EDT
SENT (0.0333s) ICMP [10.5.23.251 > 10.5.23.209 Echo request (type=8/c
ode=0) id=31901 seq=0] IP [ttl=50 id=18265 iplen=28 ]
SENT (0.0335s) TCP 10.5.23.251:53552 > 10.5.23.209:443 S ttl=58 id=17
007 iplen=44  seq=1685614607 win=1024 <mss 1460>
SENT (0.0336s) TCP 10.5.23.251:53552 > 10.5.23.209:80 A ttl=49 id=510
78 iplen=40  seq=0 win=1024
CONN (0.0354s) TCP localhost > 10.5.23.209:135 => Operation now in pr
ogress
CONN (0.0355s) TCP localhost > 10.5.23.209:135 => Connected
Nmap scan report for 10.5.23.209
Host is up (0.00024s latency).

PORT    STATE SERVICE
135/tcp open  msrpc
MAC Address: F8:B1:56:D7:2B:77 (Dell)
```

-sT packet trace
for open port

There is another example; here also use **-sT** with **packet-trace**; but now the port is open. So, you see when you sent, there are some connection responses that are coming back ok. So, because you are receiving the connections and final one is connected; after 3 way handshake the connection, it is finally connected. So, your conclusion will be the particular port is on that machine, on that host is open.

So, here you see when you do a network scan, when you try to look at vulnerabilities the first thing you look at is what are the hosts that are up and second thing you look at, is that what are the port numbers that are currently open on those hosts. Once you know that, you can try to run some exploit on those port numbers which are well known to exploit those vulnerabilities fine.

(Refer Slide Time: 11:44)



(c) TCP Stealth scan

- Basic idea:
  - Carry out port scanning while avoiding detection.
  - Try to hide themselves among normal network traffic.
  - Not to be logged (stealth).
- How it works?
  - Flag probe packets (also known as *Inverse Mapping*)
    - Response is sent back only by closed port
    - Intruder determines what services do not exist, and can infer the ones that exist.
  - Slow scan rate
    - Difficult to detect, and needs long history log.

There is some other kind of more sophisticated scan; this is called TCP stealth scan. Stealth is something as you know the term; stealth means you are hiding; no one will be able to detect you; that is the idea. So, here the idea is, you carry out port scanning, while avoiding detection; but what is the basic philosophy? How you can avoid detection? Let us try to get an idea.

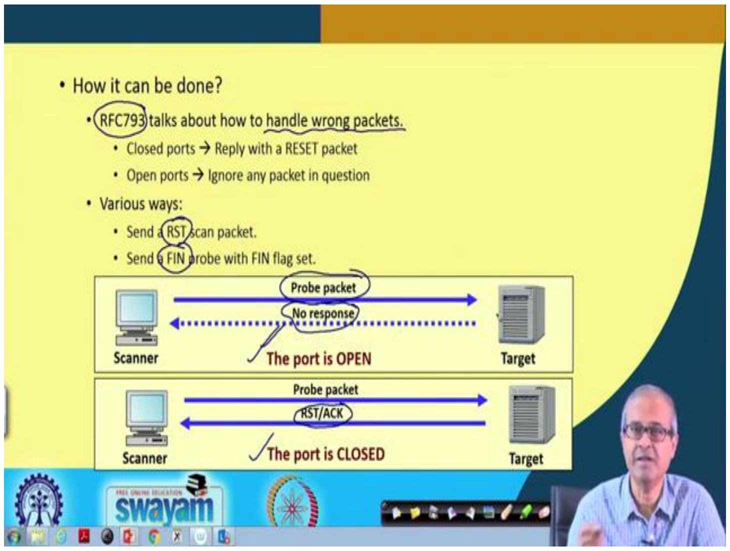
So, you are, your packet means, your means, your packets are hiding within normal network traffic. So, the firewall or the intrusion detection system, whatever you have installed on the target machine, they will not be able to distinguish your packets as some kind of malicious packets ok. And they will also not be logged; because they will, they will appear to be very harmless; that is why they will not be logged. That is how we say that they are stealthy.

And how it works? You see there are several ways in which you can do this; I am not going into this detail again. There is something called inverse mapping which is utilized; there are probe packets just like UDP scanning you just saw earlier, that the response will be send back from the target only when the port is closed; that means, the reverse, if the port is opened nothing will be send back; but if it is closed then only it is send back ok.

So, intruder determines what service do not exist; then if you take them out, you will know that what service are actually running; the ones that exists. So, this is what is? And this is difficult to detect, because you are not looking for hosts which are up or ports which are open; but on the other way around; you are looking for ports which are closed.

So, unnecessary if you are looking for ports which are closed; they will not get logged. Only if are active services things get logged. So, these typically will not be so easy to detect and needs long history log and secondly, you send this kind of packets very infrequently; only few such packets in the whole day. So, that the ideas will also not suspect that these are some malicious packets which are targeted to the hosts, very infrequent alright.

(Refer Slide Time: 14:33)


- How it can be done?
  - RFC793 talks about how to handle wrong packets.
    - Closed ports → Reply with a RESET packet
    - Open ports → Ignore any packet in question
  - Various ways:
    - Send a RST scan packet.
    - Send a FIN probe with FIN flag set.

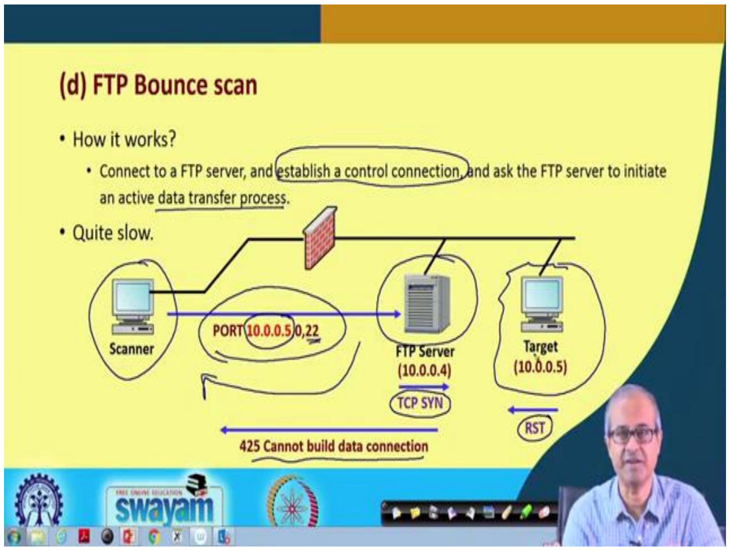
So, here we briefly try to tell you one way in which this can be done. You see every internet service that are implemented, there is a corresponding documentation available based on which the implementation is carried out. These are called RFC documents, request for comments. For example, this RFC 793, 793 is the number of that document, document number. It talks about how a host should handle wrong packets.

The idea is that this scanner, when it tries to let us say, establish a connection, it is sending a wrong packet, the probe packet. Let us say, it is sending a reset packet or a packet with the FIN flag set to 1; FIN means finish. So, these are normally not the packets with which you are initiating a connection. So, these packets are so called wrong packets to the target. So, the target, if it does not send back any response, it means it is a wrong packet; it has ignored. Which means the port is open; but according to this RFC

793, if it sends back an RST/ACK packet. This RFC says that if the port is closed and if such a packet comes that you need to terminate or reset the connection, then you conclude that the port is closed whenever the response comes back.

This is the inverse mapping; if the port is closed, then the response is coming; if the port is open, then the wrong packet is ignored; nothing is coming back ok. This is the idea behind TCP stealth scan ok.

(Refer Slide Time: 16:45)



(d) FTP Bounce scan

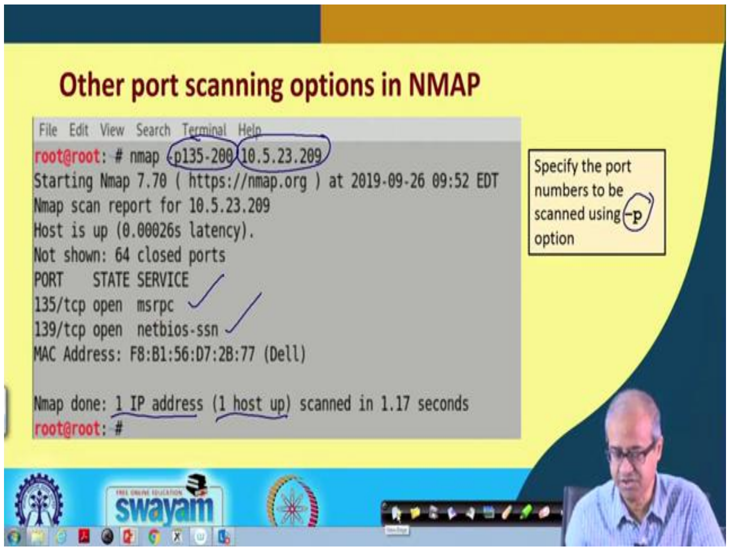
- How it works?
  - Connect to a FTP server, and establish a control connection, and ask the FTP server to initiate an active data transfer process.
- Quite slow.

There is a another way; this is called FTP Bounce scan. You see whenever we establish a connection with the FTP server to transfer a file, there are two connections that are actually established. One is called a control connection; other is called the data connection. Control connection is established to send the FTP commands, and the data connection is established to actually send the data ok. Now the idea behind FTP bounce scan is, suppose this scanner is trying to mount some kind of a discovery exercise on a target and it is utilizing an intermediate FTP server for doing that.

So, what it does; it sends back, it establishes a control connection with the FTP server and initiates a data transfer connection. So, what it does; you see FTP server let us say as an, it has an IP address 10.0.0.4 and the target has an IP address 10.0.0.5. So, when the request comes, it spoof the IP address. It changes the IP address to 10.0.0.5 which is the targets IP address. The FTP server is receiving this kind of a packet, 22 means the connection for the data; 21 is the control connection; 22 is the data connection. So, when the FTP server receives such a spoof packet, it will try to establish a data connection with 10.0.0.5; that means, this will be a TCP connection again. So, it will be sending a TCP SYN packet; it will be, but the target you see, the connection was established with the FTP server; the target does not know about it.

The target will send back a reset packet to reset the connection. It will say and when it comes back to this FTP server, FTP server will be sending back a message to this scanner that it cannot build data connection. So, the idea is that via the FTP server this scanner is getting a response back from the target and through this target that means, some packet is bouncing back and if such a response comes back, it will know, it will conclude that the target was up and running ok. So, this is called FTP bounce scan. So, on this particular port number, the target was running.

(Refer Slide Time: 19:40)



Other port scanning options in NMAP

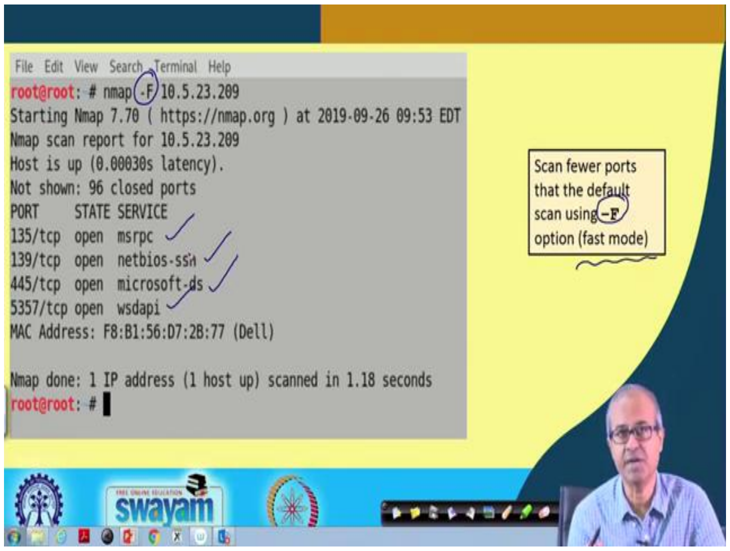
```
root@root: # nmap -p135-200 10.5.23.209
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-26 09:52 EDT
Nmap scan report for 10.5.23.209
Host is up (0.00026s latency).
Not shown: 64 closed ports
PORT    STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
MAC Address: F8:B1:56:D7:2B:77 (Dell)

Nmap done: 1 IP address (1 host up) scanned in 1.17 seconds
root@root: #
```

Specify the port numbers to be scanned using -p option

Here to I am not showing examples of these; but I am showing some other port scanning options here. Well, you can give, already saw earlier **-p** option to specify some port numbers and you can also specify a range of port numbers; like here you see **-p135-200**; so, all port numbers in this range will be scanned on this particular host. So, a scan is carried out and the final conclusion is, these two ports only are open, port number 135 and port number 139.

So, out, the other port numbers are all closed. So, it says this 1 IP address was scanned, this 1 host was up and these are the 2 services which are open on that machine. This is how you can specify a range of port numbers.

(Refer Slide Time: 20:42)



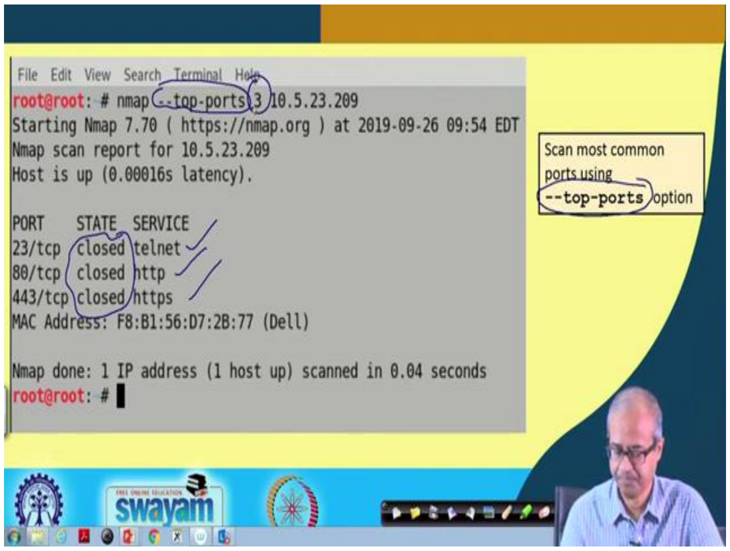
```
root@root: # nmap -F 10.5.23.209
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-26 09:53 EDT
Nmap scan report for 10.5.23.209
Host is up (0.00030s latency).
Not shown: 96 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
5357/tcp  open  wsdapi
MAC Address: F8:B1:56:D7:2B:77 (Dell)

Nmap done: 1 IP address (1 host up) scanned in 1.18 seconds
root@root: #
```

Scan fewer ports that the default scan using -F option (fast mode)

Let us look at some more examples. You can give a **-F** option; **-F** actually refers to fast mode. Fast mode means that you are not scanning all the ports on the target host; rather you are scanning fewer number of ports; the one which are most common.

So, when you scan a particular host with the **-F** option, this will also look for open ports, but not all; few number of ports will be scanned. And in that way this process will be much faster. So, for example, if you scan this, it will respond with some open ports 135, 139, 455 and 5357; these are some applications; the names of the applications are also shown here; these are presently open ok. So, this scanning with **-F** option will make the process faster.

(Refer Slide Time: 22:00)



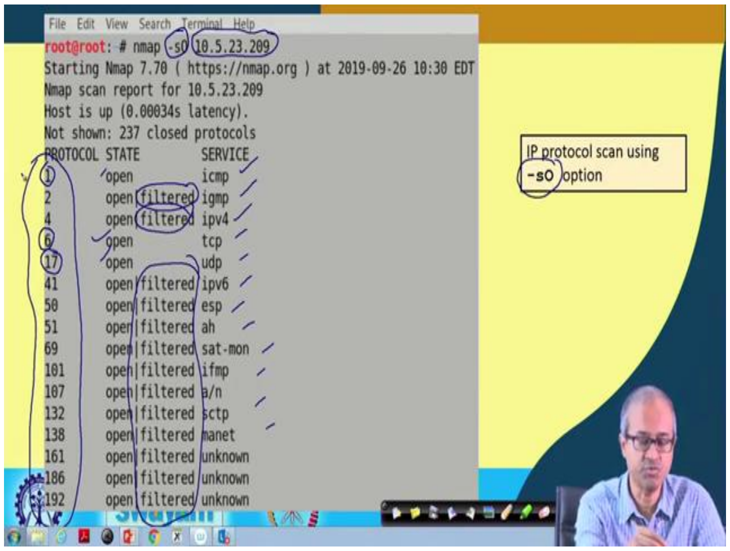
```
root@root: # nmap --top-ports 3 10.5.23.209
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-26 09:54 EDT
Nmap scan report for 10.5.23.209
Host is up (0.00016s latency).

PORT     STATE  SERVICE
23/tcp   closed telnet
80/tcp   closed http
443/tcp  closed https
MAC Address: F8:B1:56:D7:2B:77 (Dell)

Nmap done: 1 IP address (1 host up) scanned in 0.04 seconds
root@root: #
```

Scan most common ports using **--top-ports** option

Let us also look at another kind of a flag where you are using an option called **- - top-ports**. These top ports are the most commonly used ports. If you give this option with a number, the 3 most top ports, top most used ports will be scanned.

So, here you see, the 3 top ports are scanned telnet, http and https, and summery report is presented; all these 3 are closed presently on this host ok. So, you can scan some hosts with this **- - top-ports**; you can specify how many top ports, 3; you can specify 10, whatever. So, those number of ports will be scanned.

(Refer Slide Time: 23:00)



```
root@root: # nmap -sO 10.5.23.209
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-26 10:30 EDT
Nmap scan report for 10.5.23.209
Host is up (0.00034s latency).
Not shown: 237 closed protocols
PROTOCOL STATE         SERVICE
1        open          icmp
2        open|filtered igmp
4        open|filtered ipv4
6        open          tcp
17       open          udp
41       open|filtered ipv6
50       open|filtered esp
51       open|filtered ah
69       open|filtered sat-mon
101      open|filtered ifmp
107      open|filtered a/n
132      open|filtered sctp
138      open|filtered manet
161      open|filtered unknown
186      open|filtered unknown
192      open|filtered unknown
```

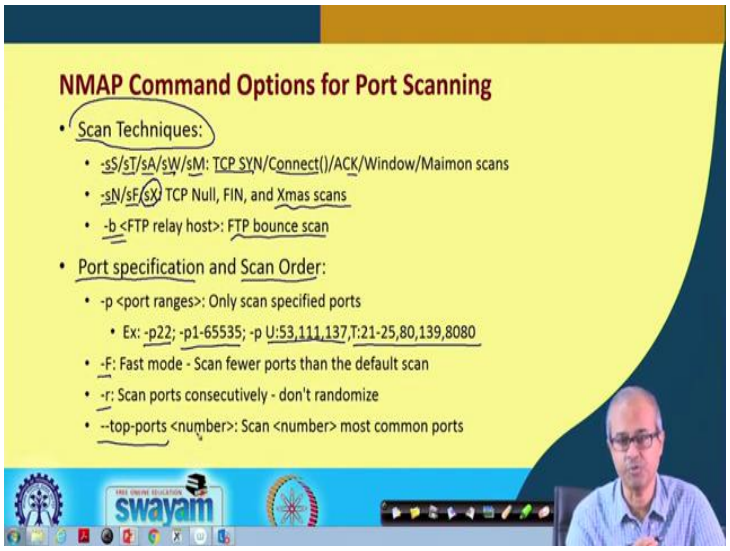
IP protocol scan using **-sO** option

Here, you are specifying something called IP protocol scan using the **-sO**, sorry **-sO** command. So, if you specify **-sO** with a particular IP address, then you specify all the protocols and you get a list like this. This is called IP protocol scan. So, what are the protocols which are currently open; some of them are open or filtered; because you are not getting back the complete response back; maybe they are filtered; but some of them are unconditionally opened. Like ICMP is open; TCP is open; UDP is open; this services are open ok.

But, there are several others which may not be open; but the protocol number, you see these are not the port number; these are protocol numbers with respect to the IP protocol, all higher level protocols that run at the transport level or higher level they have a unique number; like TCP has a number 7; UDP has a number 17. So, this ICMP has a number 1 and so on.

So, this scan is carried out based on the protocol number, not on the port number ok. So, this is a different kind of a scan where you get information about the open ports with respect to the protocol numbers. So, it depends, actually why you need this; depending on your requirement you will have to use the correct kind of scanning option.

(Refer Slide Time: 24:55)



NMAP Command Options for Port Scanning

- Scan Techniques:
  - -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
  - -sN/sF/sX: TCP Null, FIN, and Xmas scans
  - -b <FTP relay host>: FTP bounce scan
- Port specification and Scan Order:
  - -p <port ranges>: Only scan specified ports
    - Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080
  - -F: Fast mode - Scan fewer ports than the default scan
  - -r: Scan ports consecutively - don't randomize
  - --top-ports <number>: Scan <number> most common ports

So, to summarize for port scanning, this NMAP supports a number of different options; like. for instance for scan techniques, you can specify so many options **-sS, sT, sA, sW,**

sM. These are basically TCP SYN, connect, ACK, window and various other kind of scanning options; all of, all of those I have not mentioned at all.

There are **sN, sF, sX**; these stand for TCP Null, FIN, Xmas. You see this **sX** for example, stands for a Christmas scan. Well, why it is called a Christmas scan? Because you are sending a TCP packet with all these flag set, push, FIN; that means, as if your packet is glittering like a Christmas tree; when it reaches a router; router will found that so many flags are on; it will possibly allow the packet to go through thinking that it is an important packet; it is a high priority packet ok.

Push flag is also set fine; similarly you can mount the FTP bounce scan; I am not shown the example using the **-b** option. These you can do and when you specify ports, also scan order there are various options you can use. **-p** already we have seen; you can specify port ranges; you can specify a particular port; you can specify a range of port; you can specify some specific UDP and TCP ports also. Like you can specify **-p U:** these; that means, these refer to UDP ports; **T:** these refer to TCP; ports 21-25 means range, 80, 13980 these are all TCP ports.

You can specify a combination of TCP and UDP port numbers, specific port numbers also. **F**, we have seen fast mode, where you can scan fewer ports, than the default scan, where you scan everything. **-r** is consecutive scan, one by one; there is a randomization option which is default; the ports are scanned in a random order. **top-ports** also you have seen the examples with some number that how many top ports you want to scan ok.

So, with this we come to the end of this lecture, where we basically talked about different ways in which you can identify the port numbers on some hosts or a set of hosts that are opened. In the next lecture we shall be continuing with our discussion and talk about some more options that are available in NMAP for operating system discovery, some other services discovery and some common NMAP commands at the end.

Thank you.