

Scanning With NMAP

An in depth Tutorial

 St. Louis Post-Dispatch © 2000

 com

Questions? mutsonline.com

<http://mutsonline.com>

PART I : Theory

(This document is available from: [Prabhaker Mateti](#) Associate Professor at Wright State University).

Description

Port Scanning is one of the most popular reconnaissance techniques attackers use to discover services they can break into. All machines connected to a Local Area Network (LAN) or Internet run many services that listen at well-known and not so well known ports. By port scanning the attacker finds which ports are available (i.e., what service might be listening to a port). Essentially, a port scan consists of sending a message to each port, one at a time. The kind of response received indicates whether the port is used and can therefore be probed further for weakness.

Can I take legal action against port scanning?

Port scanning is like ringing the doorbell to see whether someone's at home. The police usually can't do anything about it. They have to wait until a crime is committed. The police might give it more consideration if the doorbell is repeatedly rang causing the homeowner to complain of harassment. Sometimes, if a computer system is affected too much by a port scan, one can argue that the port scan was, in fact, a denial-of-service (DoS) attack, which is usually an offense.

The various techniques in scanning are summarized below.

Port Numbers

As you know, public IP addresses are controlled by worldwide registrars, and are unique globally. Port numbers are not so controlled, but over the decades certain ports have become standard for certain services. The port numbers are unique only within a computer system. Port numbers are 16-bit unsigned numbers. The port numbers are divided into three ranges:

- ? Well Known Ports (0 - 1023)
- ? Registered Ports (1024 - 49151)
- ? Dynamic and/or Private Ports (49152 - 65535).

Well-Known Ports

Ports numbered 0 to 1023 are considered well known (also called standard ports) and are assigned to services by the IANA (Internet Assigned Numbers Authority). Here are a few samples:

echo	7/tcp	Echo
ftp-data	20/udp	File Transfer [Default Data]
ftp	21/tcp	File Transfer [Control]
ssh	22/tcp	SSH Remote Login Protocol
telnet	23/tcp	Telnet
domain	53/udp	Domain Name Server
www-http	80/tcp	World Wide Web HTTP

Non-Standard Ports

By a non-standard port, we simply mean a port whose number is higher than 1023. In this range also, several services are "standard." For example:

wins	1512/tcp	# Microsoft Windows Internet Name Service
radius	1812/udp	# RADIUS authentication protocol
yahoo	5010	# Yahoo! Messenger
x11	6000-6063/tcp	# X Window System

Some malicious programs such as Trojans and Viruses have spread so wide that there are a number of ports that if found open, usually indicate that a system may have a virus.

Simple Port Scanning Techniques

The simplest port scan tries (i.e., sends a carefully constructed packet with a chosen destination port number) each of the ports from 0 to 65535 on the victim to see which ones are open.

TCP connect(): The connect() system call provided by an OS is used to open a connection to every interesting port on the machine. If the port is listening, connect() will succeed, otherwise the port isn't reachable.

Strobe: A strobe does a narrower scan, only looking for those services the attacker knows how to exploit. The name comes from one of the original TCP scanning programs, though now virtually all scanning tools include this feature.

The ident protocol allows for the disclosure of the username of the owner of any process connected via TCP, even if that process didn't initiate the connection. So, e.g., one can connect to port 80 and then use `identd` to find out whether the HTTP server is running as root.

Stealth scan

One problem, from the perspective of the attacker attempting to scan a port, is that services listening on these ports log scans. They see an incoming connection, but no data, so an error is logged. There exist a number of stealth scan techniques to avoid this. A stealth scan is a kind of scan that is designed to go undetected by auditing tools.

Obviously, this is a race between the hacker and firewall vendors - what are considered stealth scans now may not be so in a few months once the firewall vendor becomes aware of such techniques.

Port scanners scan a host rapidly by firing off packets at different ports. So, scanning very slowly (taking a day or more) becomes a stealth technique. Another stealth scanning technique is "inverse mapping", where you try to find out all hosts on a network by generating "host unreachable" ICMP-messages for those IPs that do not exist. Since these messages may be generated by any TCP/IP packet one may send meaningless packets (e.g. RST packets sent without any previous packet).

Fragmented packets: The scanner splits the TCP header into several IP fragments. This bypasses some packet filter firewalls because they cannot see a complete TCP header that can match their filter rules. Some packet filters and firewalls do queue all IP fragments, but many networks cannot afford the performance loss caused by the queuing.

SYN scanning: This technique is also called *half-open* scanning, because a TCP connection is not completed. A SYN packet is sent (as if we are going to open a connection), and the target host responds with a SYN+ACK, this indicates the port is listening, and an RST indicates a non-listener. The server process is never informed by the TCP layer because the connection did not complete.

FIN scanning: The typical TCP scan attempts to open connections (at least part way). Another technique sends erroneous packets at a port, expecting that open listening ports will send back different error messages than closed ports. The scanner sends a FIN packet, which should close a connection that is open. Closed ports reply to a FIN packet with a RST. Open ports, on the other hand, ignore the packet in question. This is required TCP behavior. If no service is listening at the target port, the operating system will generate an error message. If a service is listening, the operating system will silently drop the incoming packet. Therefore, silence indicates the presence of a service at the port. However, since packets can be dropped accidentally on the wire or blocked by firewalls, this isn't a very effective scan.

Other techniques that have been used consist of **XMAS scans** where all flags in the TCP packet are set, or **NULL scans** where none of the bits are set. However, different operating systems respond differently to these scans, and it becomes important to identify the OS and even its version and patch level.

Bounce Scans

The ability to hide their tracks is important to attackers. Therefore, attackers scour the Internet looking for systems they can bounce their attacks through.

FTP bounce scanning takes advantage of a vulnerability of the FTP protocol itself. It requires support for proxy ftp connections. This bouncing through an FTP server hides where the attacker comes from. This technique is similar to IP spoofing in that it hides where the attacker comes from. For example, *evil.com* establishes a control connection to the FTP server-PI (protocol interpreter) of *target.com*, then requests that the server-PI initiate an active server-DTP (data transfer process) to send a file anywhere on the Internet.

A port scanner can exploit this to scan TCP ports from a proxy ftp server. Thus you could connect to an FTP server behind a firewall, and then scan ports that are more likely to be blocked (e.g., port 139). If the ftp server allows reading from and writing to a directory (such as `/incoming`), you can send arbitrary data to ports that you do find open.

The advantages to this approach are obvious (harder to trace, potential to bypass firewalls). The main disadvantages are that it is slow, and that many FTP server implementations have finally disabled the proxy "feature".

SOCKS Allows almost any protocol to be tunneled through the intermediate machine. As a result, attackers probing for SOCKS is common scan seen on the Internet.

HTTP proxy: Most web servers support proxying so that all web traffic can be directed to a single server for filtering as well as caching to improve performance. A lot of these servers are misconfigured to allow proxying of any request from the Internet, allowing attackers to relay attacks against web sites through a third party. Probes for HTTP proxies are one of the more common scans seen today.

IRC BNC: Attackers love to hide their IRC identities by bouncing their connections through other machines. A particular program called "BNC" is used for this purpose on compromised machines.

UDP Scanning

Port scanning usually means scanning for TCP ports, which are connection-oriented and therefore give good feedback to the attacker. UDP responds in a different manner. In order to find UDP ports, the attacker generally sends empty UDP datagrams. If the port is listening, the service should send back an error message or ignore the incoming datagram. If the port is closed, then most operating systems send back an "ICMP Port Unreachable" message. Thus, you can find out if a port is NOT open, and by exclusion determine which ports are open. Neither UDP packets, nor the ICMP errors are guaranteed to arrive, so UDP scanners of this sort must also implement retransmission of packets that appear to be lost (or you will get a bunch of false positives). Also, this scanning technique is slow because of compensation for machines that implement the suggestions of RFC 1812 and limit ICMP error message rate. For example, a kernel may limit destination unreachable message generation to 80 per 4 seconds, with a 1/4 second penalty if that is exceeded.

Some people think UDP scanning is pointless - not so. Sometimes for example, Rpcbind can be found hiding on an undocumented UDP port somewhere above 32770. So it doesn't matter that port 111 is blocked by the firewall. But can you find which of the more than 30,000 high ports it is listening on? With a UDP scanner you can.

ICMP Scanning:

This isn't really port scanning, since ICMP does not have a port abstraction. But it is sometimes useful to determine what hosts in a network are up by pinging them all. ICMP scanning can be done in parallel, so it can be quite fast.

Fingerprinting an OS:

The last scanning method is called Fingerprinting. Fingerprinting is the technique of interpreting the responses of a system in order to figure out what it is. Unusual combinations of data are sent to the system in order to trigger these responses. Systems respond the same with correct data, but they rarely respond the same way for wrong data.

This document is available from: [Prabhaker Mateti](#) Associate Professor at Wright State University.

PART II : Practice

NMAP Scanning options:

- sS TCP SYN stealth port scan (default if privileged (root))
- sT TCP connect() port scan (default for unprivileged users)
- sU UDP port scan
- sP ping scan (Find any reachable machines)
- sF,-sX,-sN Stealth FIN, Xmas, or Null scan (experts only)

Some Common Options (none are required, most can be combined):

- O Use TCP/IP fingerprinting to guess remote operating system
- p <range> Ports to scan. Example range: '1-1024,1080,6666,31337'
- F Only scans ports listed in nmap-services
- v Verbose. Its use is recommended. Use twice for greater effect.
- P0 Don't ping hosts (needed to scan www.microsoft.com and others)
- D decoy_host1,decoy2[...] Hide scan using many decoys
- T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane> General timing policy
- n/-R Never do DNS resolution/Always resolve [default: sometimes resolve]
- iL <inputfile> Get targets from file; Use '-' for stdin
- S <your_IP>/-e <devicename> Specify source address or network interface

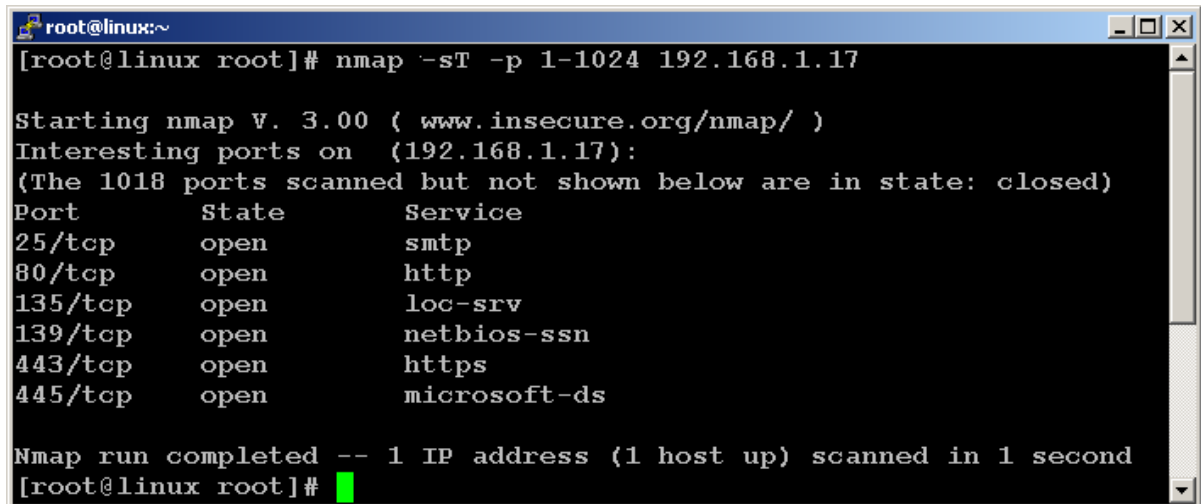
- Uhm...um yes...quite a lot of options, and at first it may seem a bit overwhelming, but once you get started, it's not so bad.
- Download NMAP CLI version (for Windows) from **nutsonline.com** or any other trusted source to test out these scans. If you prefer to use Linux, you may do so, the command line switches are the same.
- You should download some sort of IDS (Like BlackICE Defender) to check scan logs.

Always remember that scanning a computer that is not yours is illegal!

Step By Step Scans

1. Lets start with a normal TCP Connect Scan on Machine 192.168.1.17, ports 1-1024.

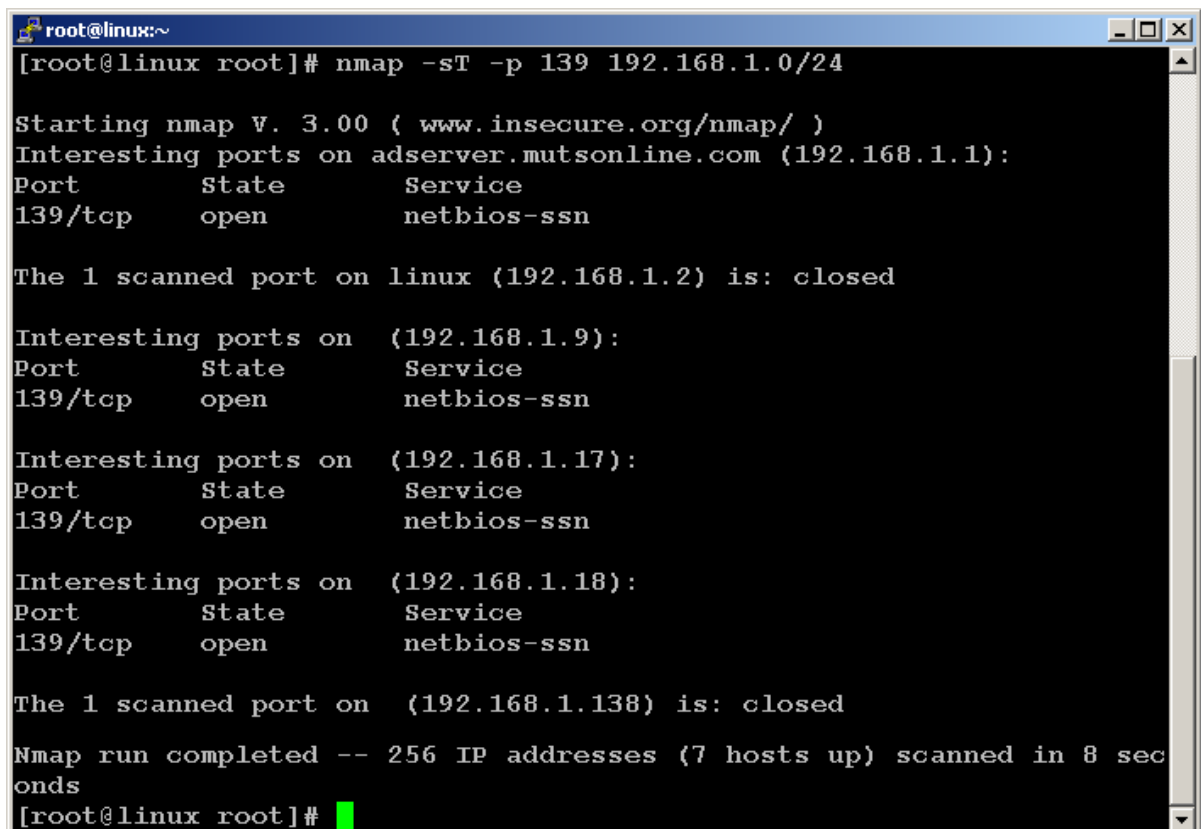
```
nmap -sT -p 1-1024 192.168.1.17
```

A terminal window showing the execution of an nmap scan on 192.168.1.17. The output lists several open ports with their corresponding services: 25/tcp (smtp), 80/tcp (http), 135/tcp (loc-srv), 139/tcp (netbios-ssn), 443/tcp (https), and 445/tcp (microsoft-ds). The scan completed in 1 second.

```
root@linux:~  
[root@linux root]# nmap -sT -p 1-1024 192.168.1.17  
  
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )  
Interesting ports on (192.168.1.17):  
(The 1018 ports scanned but not shown below are in state: closed)  
Port      State      Service  
25/tcp    open       smtp  
80/tcp    open       http  
135/tcp   open       loc-srv  
139/tcp   open       netbios-ssn  
443/tcp   open       https  
445/tcp   open       microsoft-ds  
  
Nmap run completed -- 1 IP address (1 host up) scanned in 1 second  
[root@linux root]#
```

2. Now we'll scan a whole subnet for machines with open port 139 (Netbios):

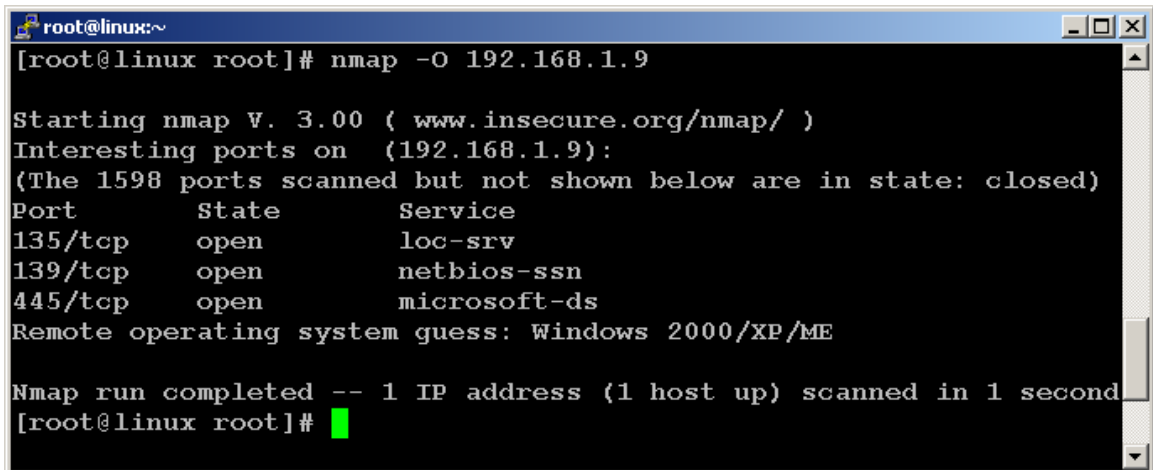
```
nmap -sT -p 139 192.168.1.0/24
```

A terminal window showing the execution of an nmap scan on the subnet 192.168.1.0/24. The output shows that port 139 (netbios-ssn) is open on several hosts: adserver.mutsonline.com (192.168.1.1), 192.168.1.9, 192.168.1.17, and 192.168.1.18. The scan completed in 8 seconds.

```
root@linux:~  
[root@linux root]# nmap -sT -p 139 192.168.1.0/24  
  
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )  
Interesting ports on adserver.mutsonline.com (192.168.1.1):  
Port      State      Service  
139/tcp   open       netbios-ssn  
  
The 1 scanned port on linux (192.168.1.2) is: closed  
  
Interesting ports on (192.168.1.9):  
Port      State      Service  
139/tcp   open       netbios-ssn  
  
Interesting ports on (192.168.1.17):  
Port      State      Service  
139/tcp   open       netbios-ssn  
  
Interesting ports on (192.168.1.18):  
Port      State      Service  
139/tcp   open       netbios-ssn  
  
The 1 scanned port on (192.168.1.138) is: closed  
  
Nmap run completed -- 256 IP addresses (7 hosts up) scanned in 8 seconds  
[root@linux root]#
```

3. Lets try guessing the operating system of 192.168.1.9:

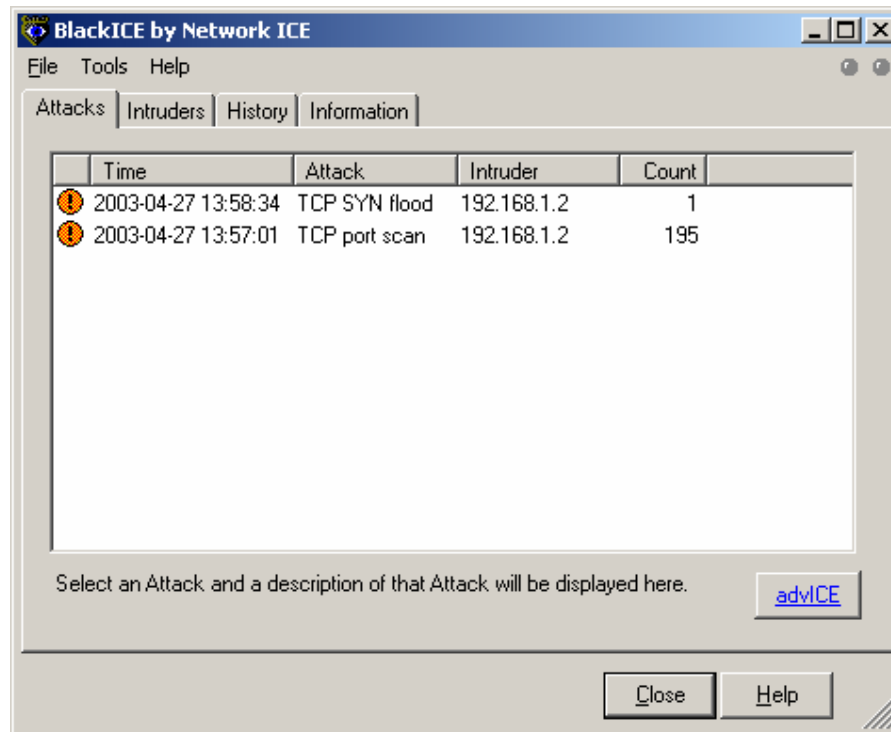
nmap -O 192.168.1.9



```
root@linux:~  
[root@linux root]# nmap -O 192.168.1.9  
  
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )  
Interesting ports on (192.168.1.9):  
(The 1598 ports scanned but not shown below are in state: closed)  
Port      State      Service  
135/tcp   open      loc-srv  
139/tcp   open      netbios-ssn  
445/tcp   open      microsoft-ds  
Remote operating system guess: Windows 2000/XP/ME  
  
Nmap run completed -- 1 IP address (1 host up) scanned in 1 second  
[root@linux root]#
```

We see it's a Windows 2000/XP/ME machine.

4. **Now for the more interesting options NMAP has to offer.** Open BlackIce 2.5 IDS on your scanned (target) computer and try a normal scan on this computer. Scan the machine and see what logs are created on the IDS.

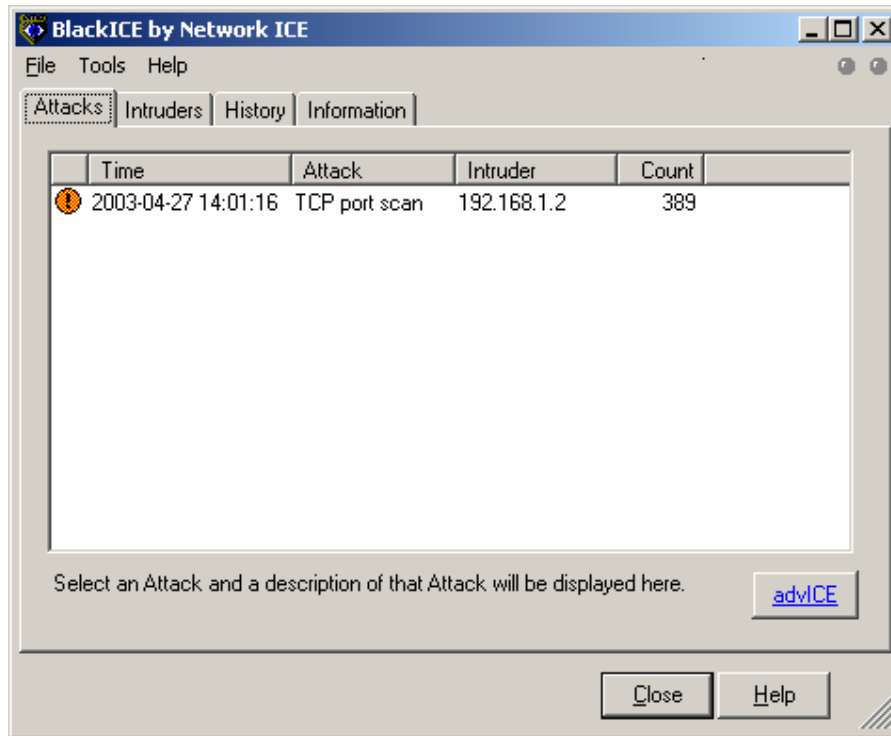


We can see that our scan was detected by Blackice.

(Our attacking computer has the IP 192.168.1.2).

5. Lets try a Stealth Scan and see the results:

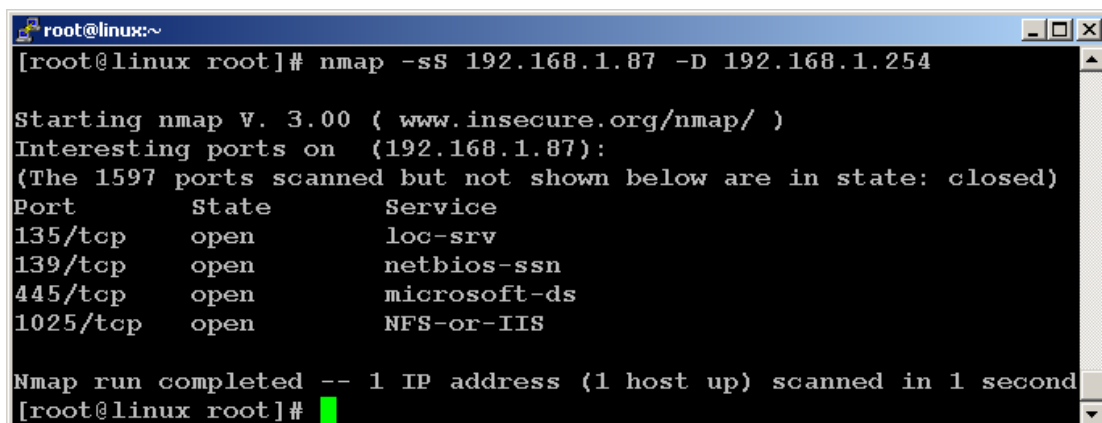
```
nmap -sS 192.168.1.87
```



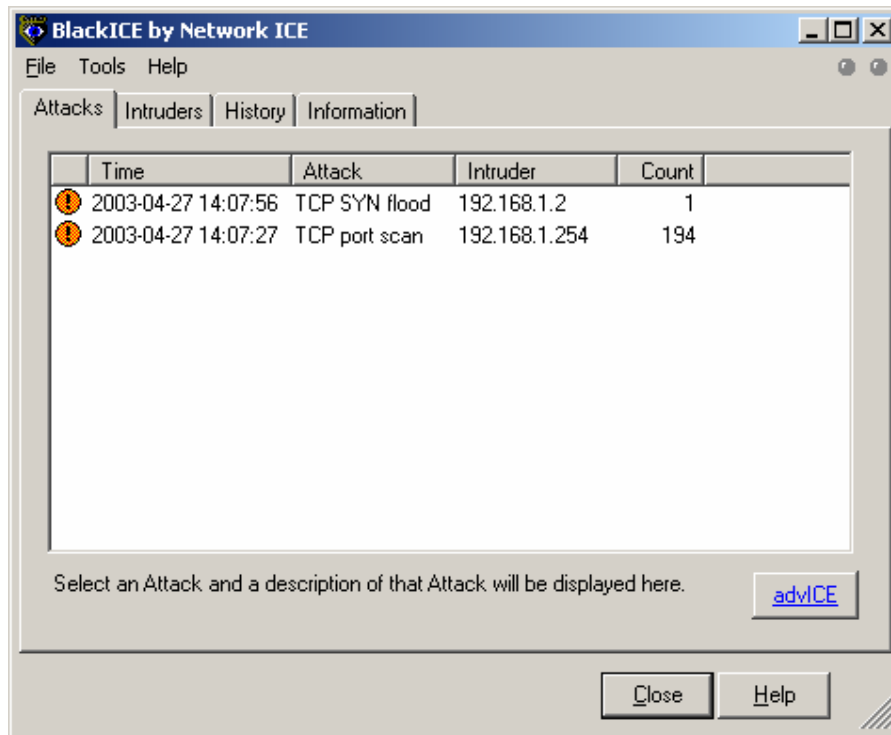
Hmm...it seems that our "Stealth Scan" is not as stealthy as it seems...Most IDS can identify "Stealth Scans", so don't rely on this!

6. Lets try a "Decoy Scan":

```
nmap -sS 192.168.1.87 -D 192.168.1.254
```



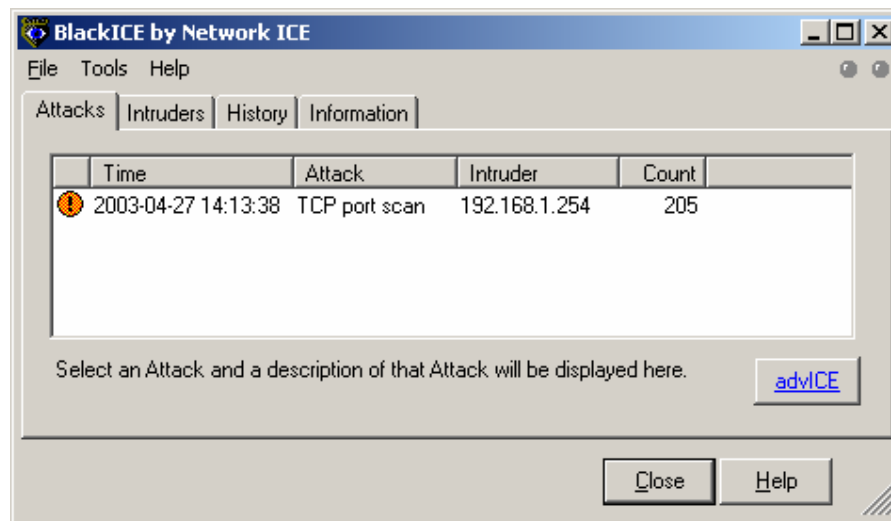
With this scan, we use an additional "Source IP address" in order to confuse and blur our tracks. We actually spoofed a scan from a nonexistent machine, 192.168.1.254. Let's look at what BlackIce has to say about this.



Here we can see that BlackIce logged both IP's, real and spoofed. In fact, most of the scans look like they've come from the spoofed address...Interesting.

- Let's try a completely spoofed Scan. This means that we are sending out packets with a modified source IP address. This will trick the IDS system into thinking the scan is from a different IP than ours. **However, the packets will not be returned to us, so we won't actually get any information out of this scan.** It's good for framing or confusing logs, or otherwise being naughty ☺

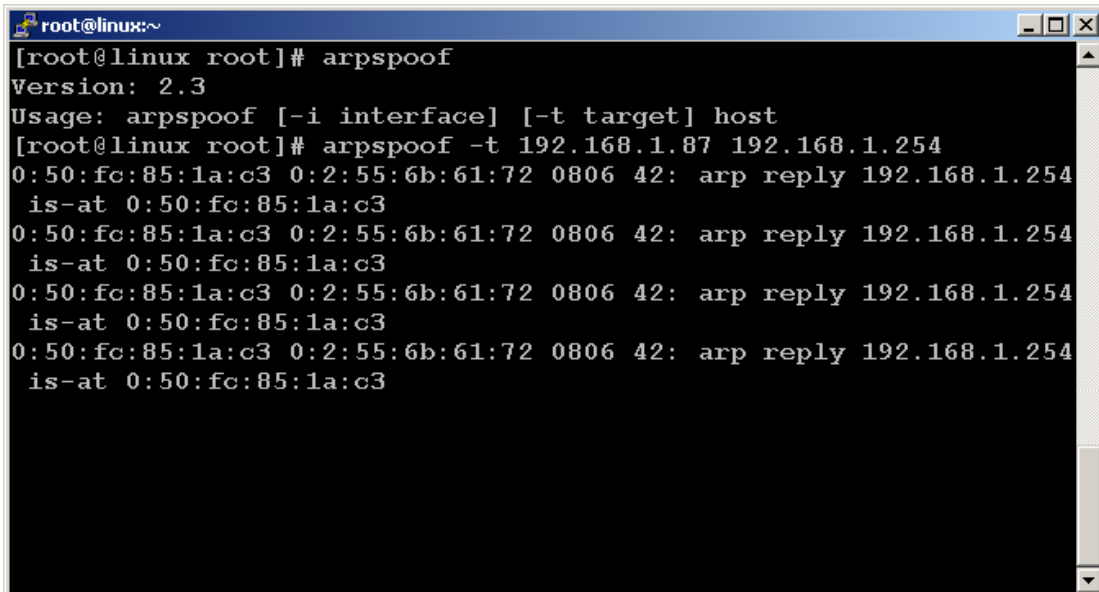
```
nmap -sS -P0 -e eth0 -S 192.168.1.254 192.168.1.87
```



We can see that the IDS system has logged the spoofed address, and not our real address.

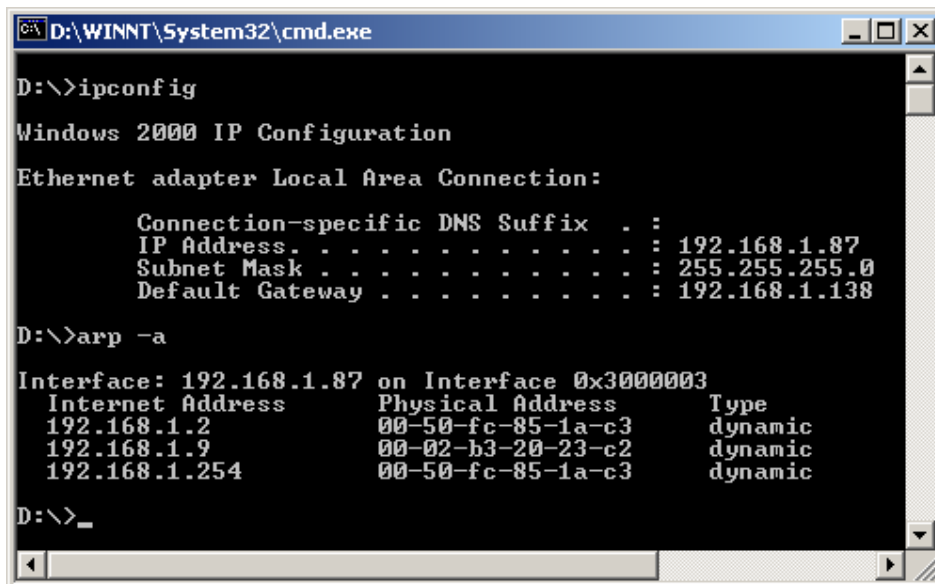
8. By combining NMAP with ARP spoofing, we can poison our victim computers' ARP Cache into thinking that 192.168.1.254 is us, thereby receiving the sent packets with the spoofed address:

arpspoof -t 192.168.1.87 192.168.1.254



```
root@linux:~  
[root@linux root]# arpspoof  
Version: 2.3  
Usage: arpspoof [-i interface] [-t target] host  
[root@linux root]# arpspoof -t 192.168.1.87 192.168.1.254  
0:50:fc:85:1a:c3 0:2:55:6b:61:72 0806 42: arp reply 192.168.1.254  
is-at 0:50:fc:85:1a:c3  
0:50:fc:85:1a:c3 0:2:55:6b:61:72 0806 42: arp reply 192.168.1.254  
is-at 0:50:fc:85:1a:c3  
0:50:fc:85:1a:c3 0:2:55:6b:61:72 0806 42: arp reply 192.168.1.254  
is-at 0:50:fc:85:1a:c3  
0:50:fc:85:1a:c3 0:2:55:6b:61:72 0806 42: arp reply 192.168.1.254  
is-at 0:50:fc:85:1a:c3
```

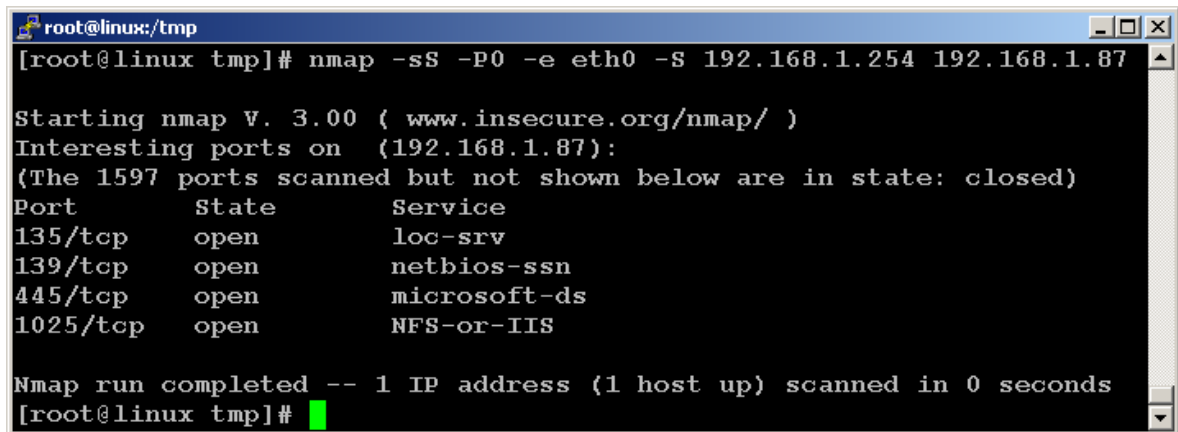
9. If we take a look at our attacked computers' ARP Cache we will see what this is doing:



```
D:\>ipconfig  
Windows 2000 IP Configuration  
Ethernet adapter Local Area Connection:  
Connection-specific DNS Suffix . :  
IP Address. . . . . : 192.168.1.87  
Subnet Mask . . . . . : 255.255.255.0  
Default Gateway . . . . . : 192.168.1.138  
D:\>arp -a  
Interface: 192.168.1.87 on Interface 0x3000003  
Internet Address Physical Address Type  
192.168.1.2 00-50-fc-85-1a-c3 dynamic  
192.168.1.9 00-02-b3-20-23-c2 dynamic  
192.168.1.254 00-50-fc-85-1a-c3 dynamic  
D:\>_
```

We can see that the attacked computer has a thinks that 192.168.1.254 has the MAC address of 192.168.1.2, therefore all Spoofed scanning traffic is now actually being redirected to 192.168.1.2.

10. We can see that the spoofed (with the use of ARPSpoof) scan now gives us results:

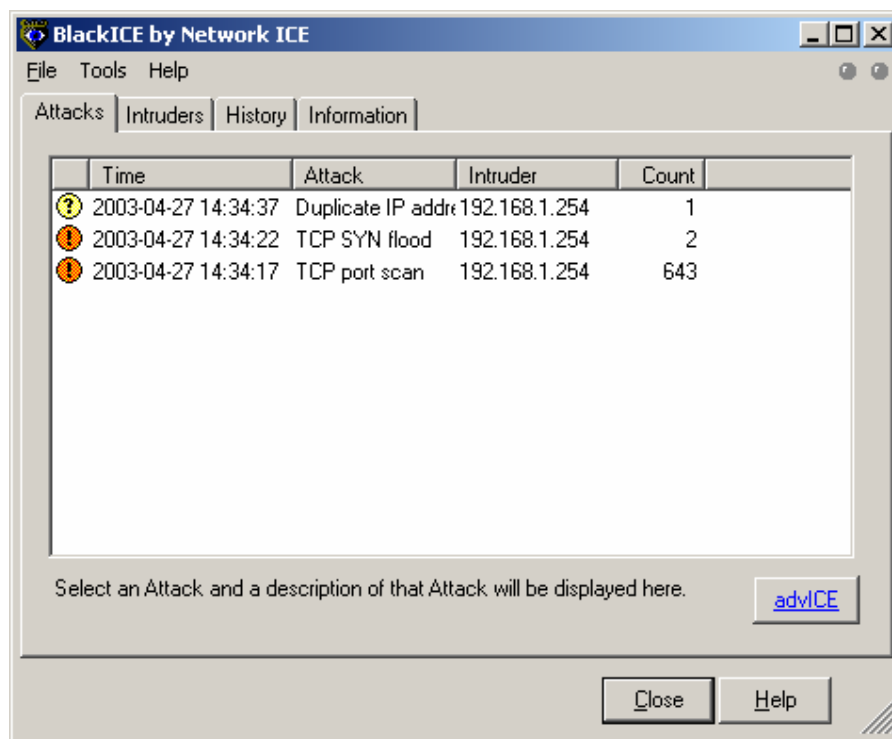


```
root@linux:/tmp
[root@linux tmp]# nmap -sS -P0 -e eth0 -S 192.168.1.254 192.168.1.87

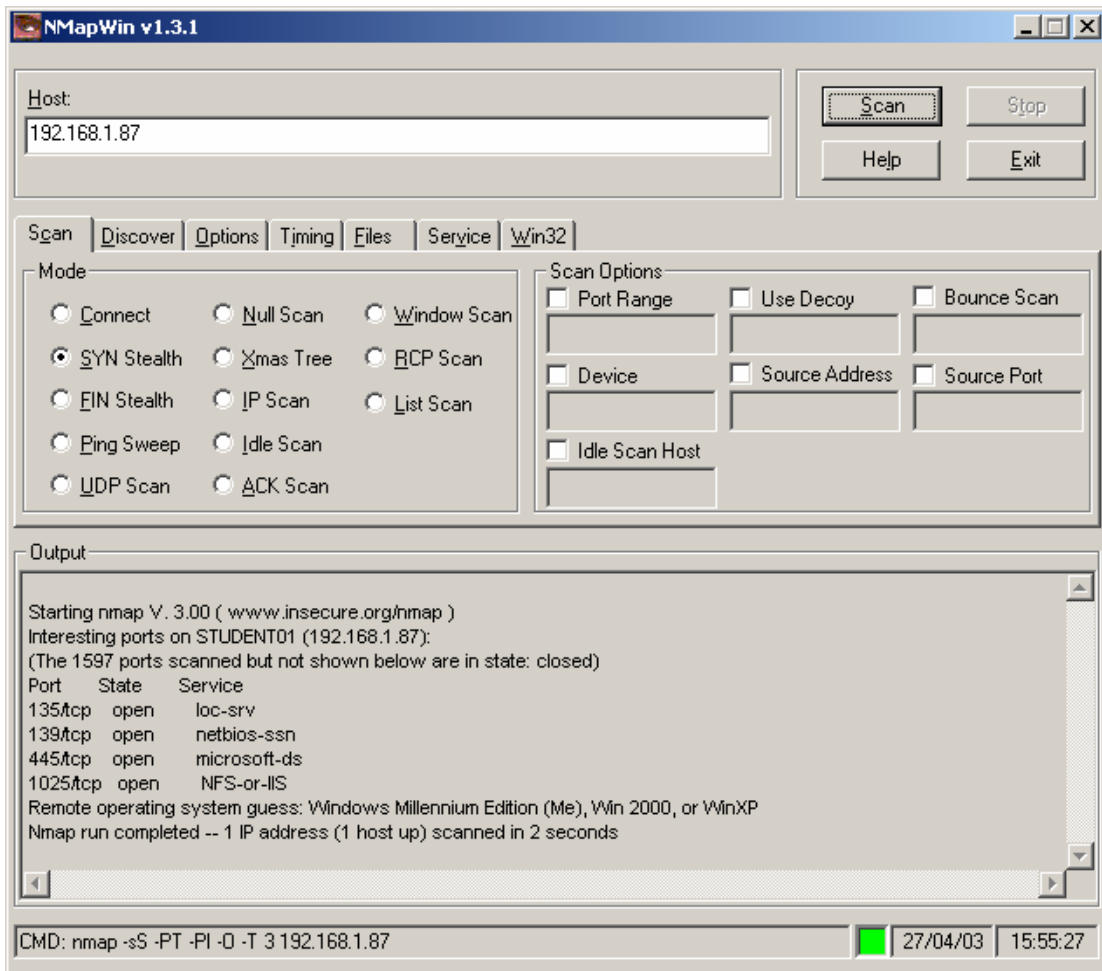
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on (192.168.1.87):
(The 1597 ports scanned but not shown below are in state: closed)
Port      State      Service
135/tcp   open       loc-srv
139/tcp   open       netbios-ssn
445/tcp   open       microsoft-ds
1025/tcp  open       NFS-or-IIS

Nmap run completed -- 1 IP address (1 host up) scanned in 0 seconds
[root@linux tmp]#
```

The IDS logs a spoofed address of 192.168.1.254 which doesn't even exist!



All these sophisticated NMAP scans have earned it the reputation of the best scanner out there. NMAP for Windows platforms can also be found either with a graphical user interface or a CLI version at http://www.insecure.org/nmap/nmap_download.html.



The End.