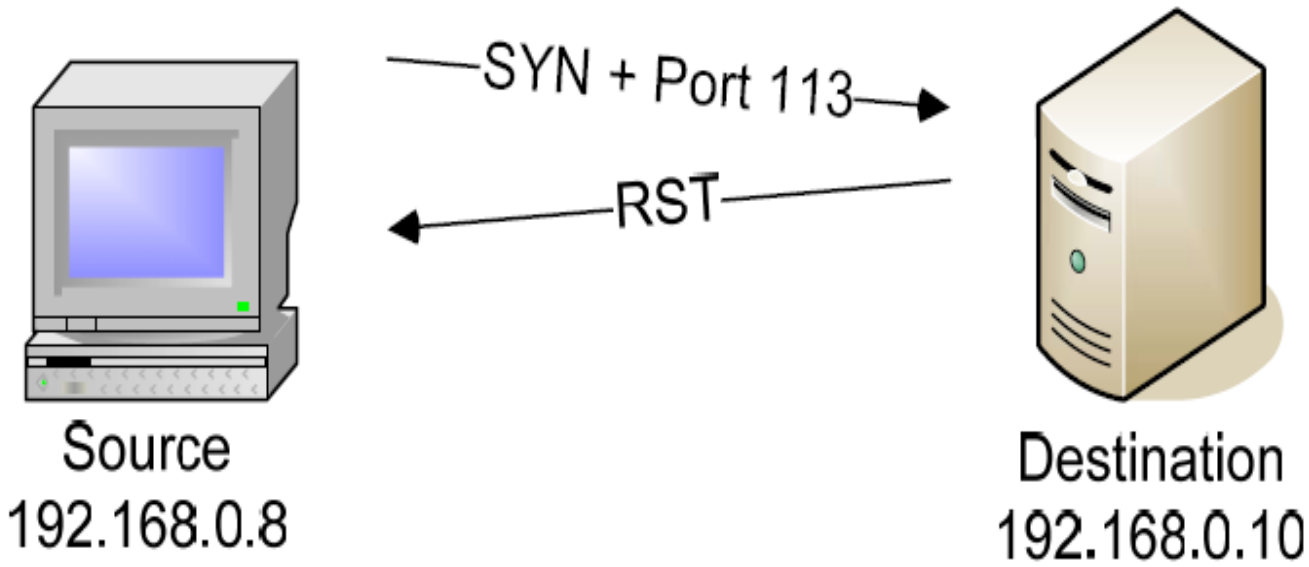
Nmap Scan	Command Syntax	Requires Privileged Access	Identifies TCP Ports	Identifies UDP Ports
TCP SYN Scan	`-sS`	YES	YES	NO
TCP connect() Scan	`-sT`	NO	YES	NO
FIN Scan	`-sF`	YES	YES	NO
Xmas Tree Scan	`-sX`	YES	YES	NO
Null Scan	`-sN`	YES	YES	NO
Ping Scan	`-sP`	NO	NO	NO
Version Detection	`-sV`	NO	NO	NO
UDP Scan	`-sU`	YES	NO	YES
IP Protocol Scan	`-sO`	YES	NO	NO
ACK Scan	`-sA`	YES	YES	NO
Window Scan	`-sW`	YES	YES	NO
RPC Scan	`-sR`	NO	NO	NO
List Scan	`-sL`	NO	NO	NO
Idlescan	`-sI`	YES	YES	NO
FTP Bounce Attack	`-b`	NO	YES	NO

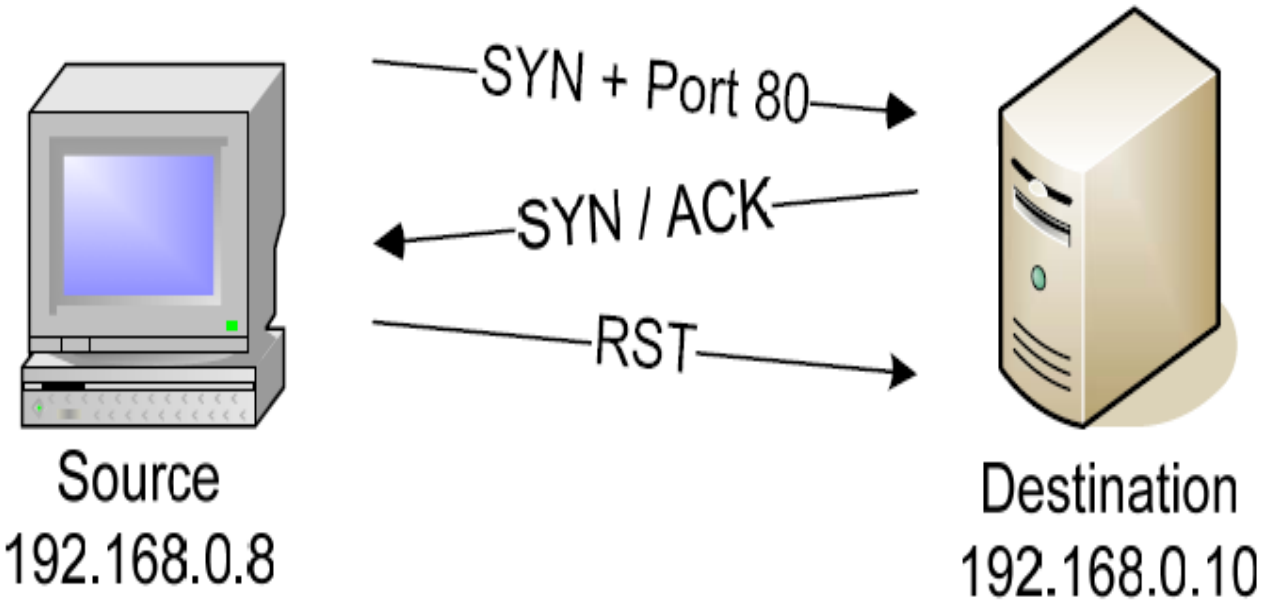
TCP SYN Scan Operation

Most of the ports queried during the TCP SYN scan will probably be closed. These closed port responses to the TCP SYN frame will be met with a RST frame from the destination station.




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.10]  TCP: D=113 S=57283 SYN SEQ=2360927338 LEN=0 WIN=3072
[192.168.0.10]  [192.168.0.8}   TCP: D=57283 S=113 RST ACK=2360927339 WIN=0
```

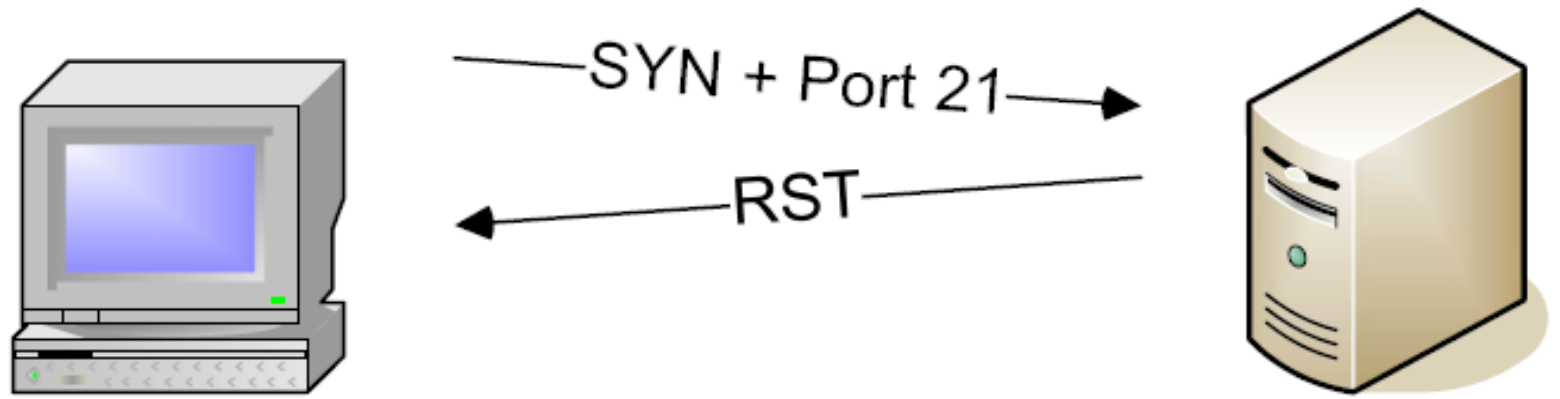
If nmap receives an acknowledgment to a SYN request, then the port is open. Nmap then sends an RST to reset the session, and the handshake is never completed.




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.10]  TCP: D=80 S=57283 SYN SEQ=2360927338 LEN=0 WIN=3072
[192.168.0.10]  [192.168.0.8]   TCP: D=57283 S=80 SYN ACK=2360927339 SEQ=1622899389 LEN=0 WIN=65535
[192.168.0.8]   [192.168.0.10]  TCP: D=80 S=57283 RST WIN=0
```

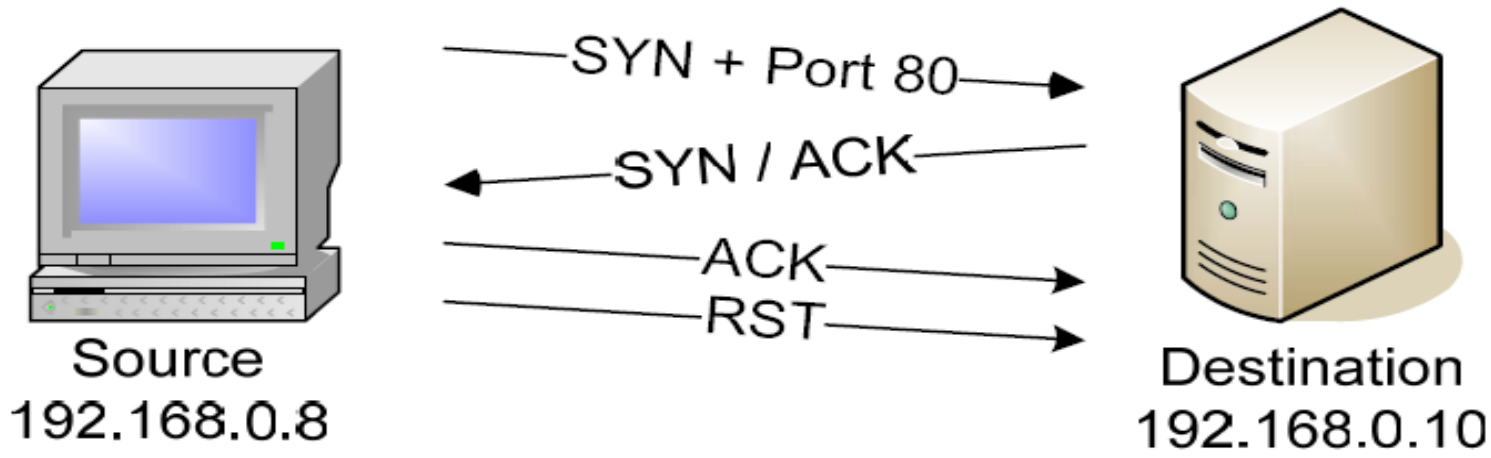
TCP connect() Scan Operation

The TCP connect() scan to a closed port looks exactly like the TCP SYN scan:




```
Source           Destination    Summary
-------------------------------------------------------------------------------
[192.168.0.8]    [192.168.0.10] TCP: D=21 S=41441 SYN SEQ=3365539736 LEN=0 WIN=5840
[192.168.0.10]   [192.168.0.8]  TCP: D=41441 S=21 RST ACK=3365539737 WIN=0
```

A scan to an open port results in a different traffic pattern than the TCP SYN scan:




```
Source           Destination    Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.10]  TCP: D=80 S=49389 SYN SEQ=3362197786 LEN=0 WIN=5840
[192.168.0.10] [192.168.0.8]    TCP: D=49389 S=80 SYN ACK=3362197787 SEQ=58695210 LEN=0 WIN=65535
[192.168.0.8]   [192.168.0.10]  TCP: D=80 S=49389 ACK=58695211 WIN<<2=5840
[192.168.0.8]   [192.168.0.10]  TCP: D=80 S=49389 RST ACK=58695211 WIN<<2=5840
```

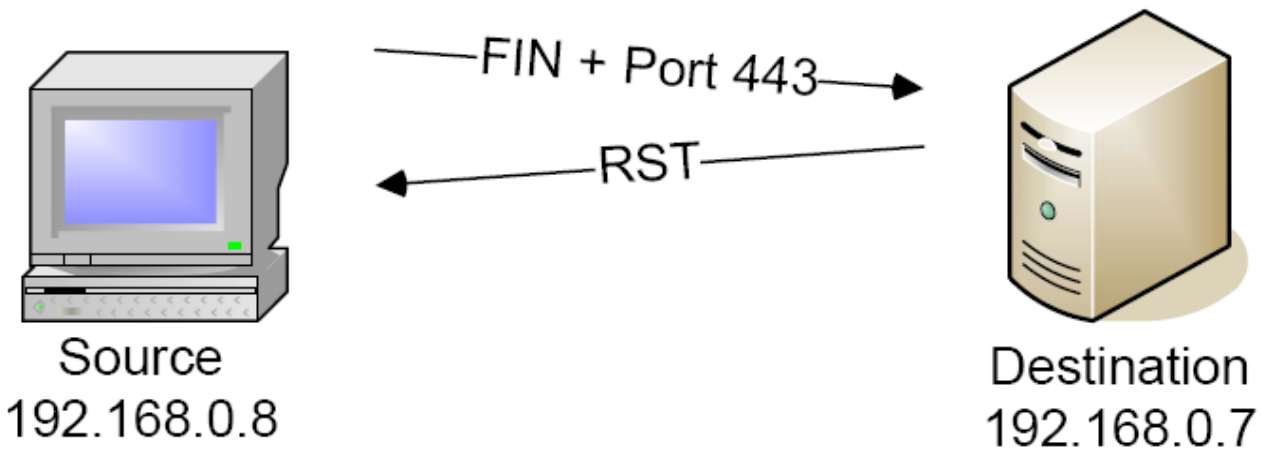
As the trace file excerpt shows, the TCP connect() scan completed the TCP three-way handshake and then immediately sent a reset (RST) packet to close the connection.

Unlike the TCP SYN scan, the nmap output shows that very few raw packets were required for the TCP connect() process to complete:

The FIN Scan (`-sF`)

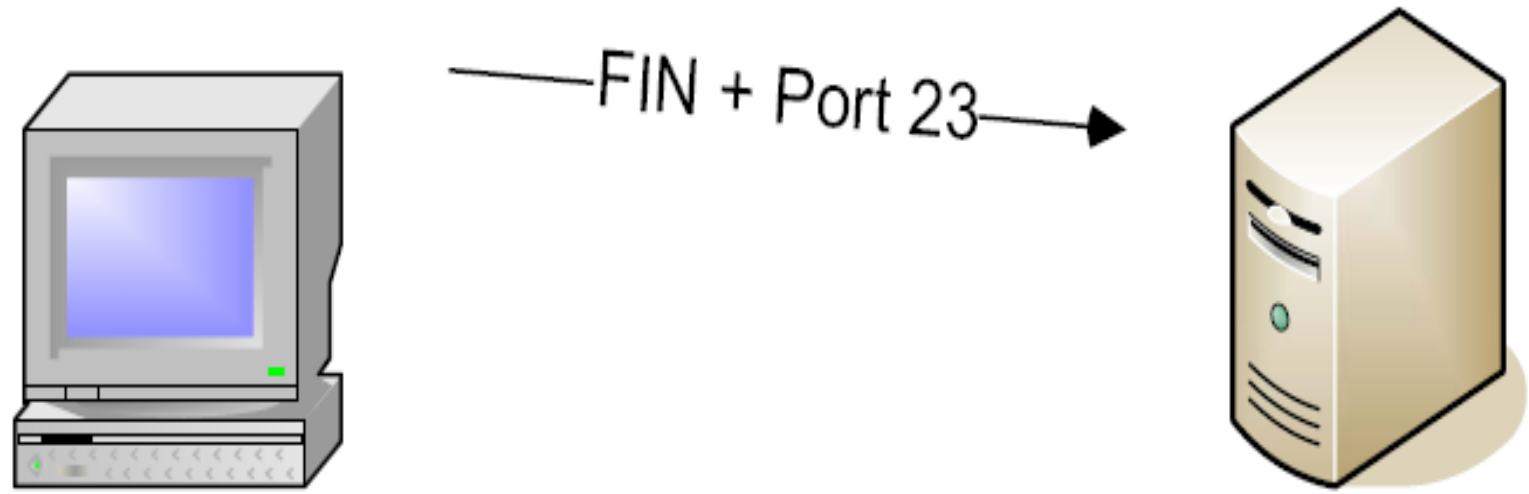
The FIN scan's "stealth" frames are unusual because they are sent to a device without first going through the normal TCP handshaking. If a TCP session isn't active, the session certainly can't be formally closed!

In this FIN scan, TCP port 443 is closed so the remote station sends a RST frame response to the FIN packet:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]  [192.168.0.7]  TCP: D=443 S=62178 FIN SEQ=3532094343 LEN=0 WIN=2048
[192.168.0.7]  [192.168.0.8]  TCP: D=62178 S=443 RST ACK=3532094343 WIN=0
```

If a port is open on a remote device, no response is received to the FIN scan:

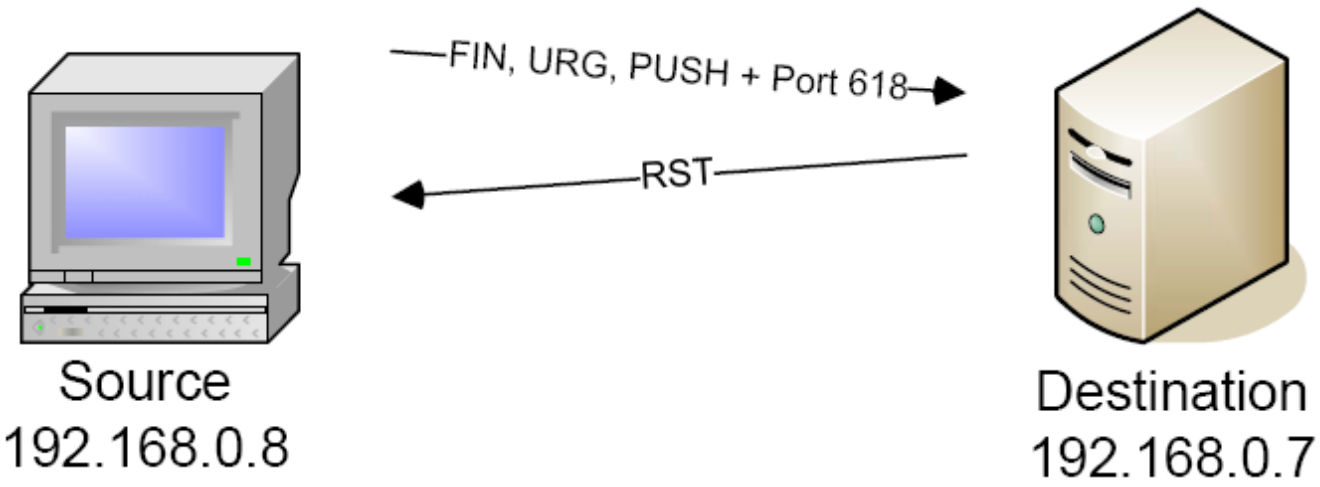



```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8] [192.168.0.7] TCP: D=23 S=62178 FIN SEQ=3532094343 LEN=0 WIN=2048
```

The Xmas Tree Scan (`-sX`)

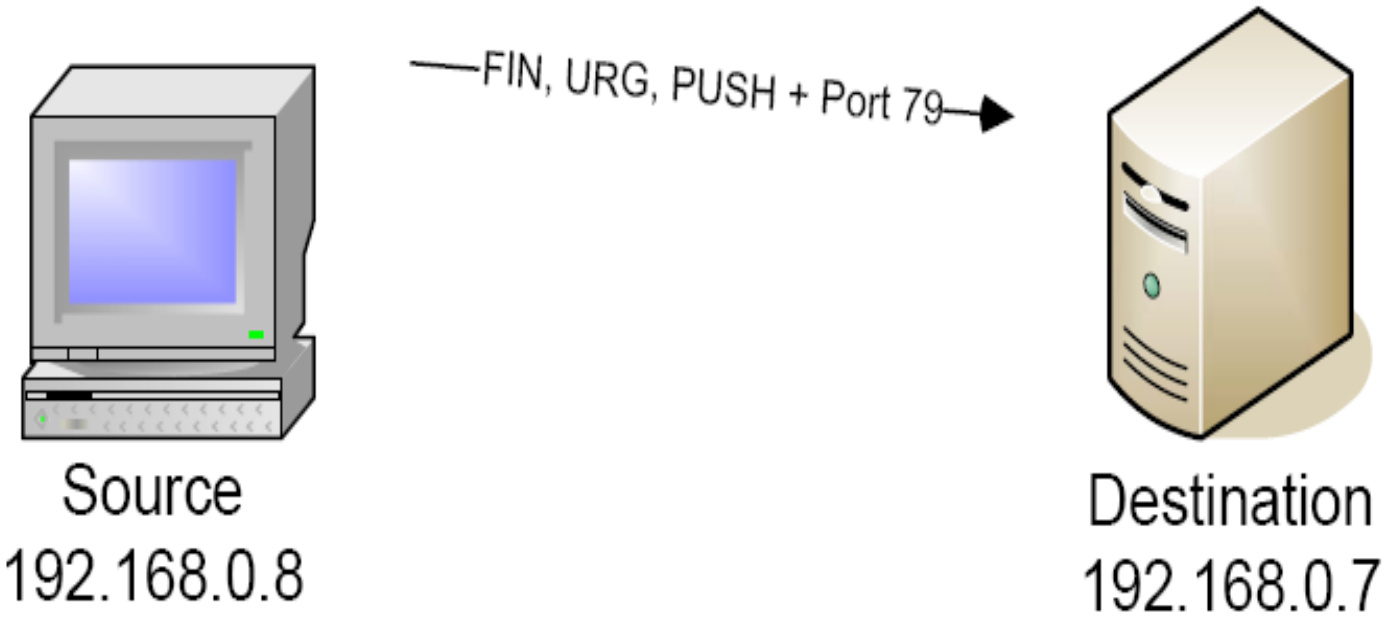
The Xmas tree scan sends a TCP frame to a remote device with the URG, PUSH, and FIN flags set. This is called a Xmas tree scan because of the alternating bits turned on and off in the flags byte (`00101001`), much like the lights of a Christmas tree.

A closed port responds to a Xmas tree scan with a RST:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.7]   TCP: D=618 S=36793 FIN URG PUSH SEQ=3378228596 LEN=0 WIN=1024
[192.168.0.7]   [192.168.0.8]   TCP: D=36793 S=618 RST ACK=3378228596 WIN=0
```

Similar to the FIN scan, an open port on a remote station is conspicuous by its silence:

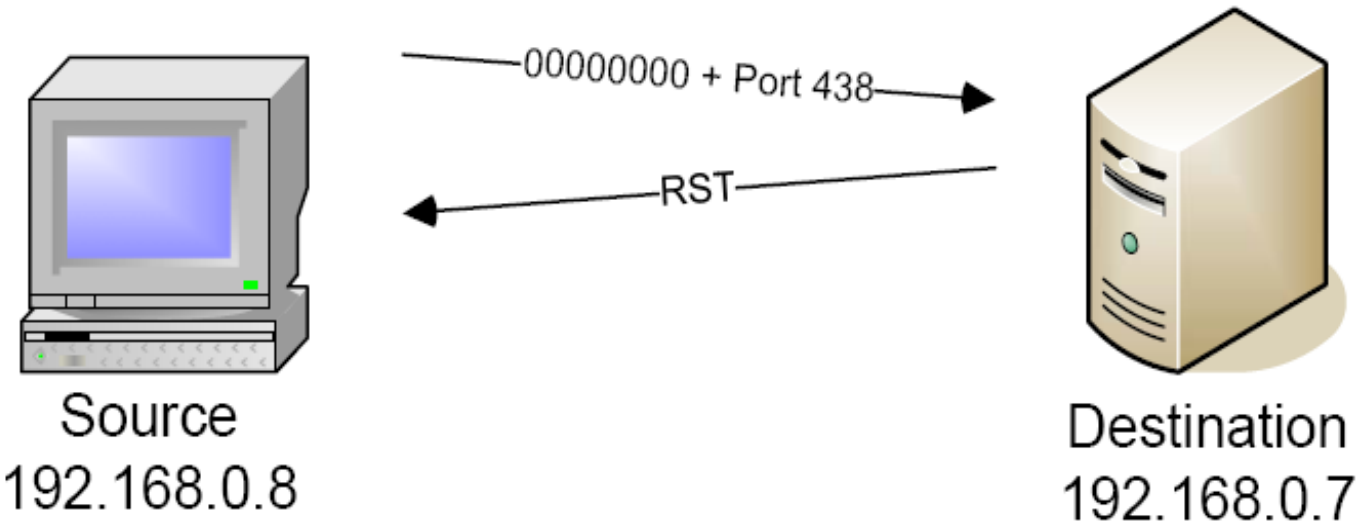



```
Source          Destination    Summary
-------------------------------------------------------------------------------
[192.168.0.8] [192.168.0.7]  TCP: D=79 S=36793 FIN URG PUSH SEQ=3378228596 LEN=0 WIN=2048
```

The Null Scan (`-sN`)

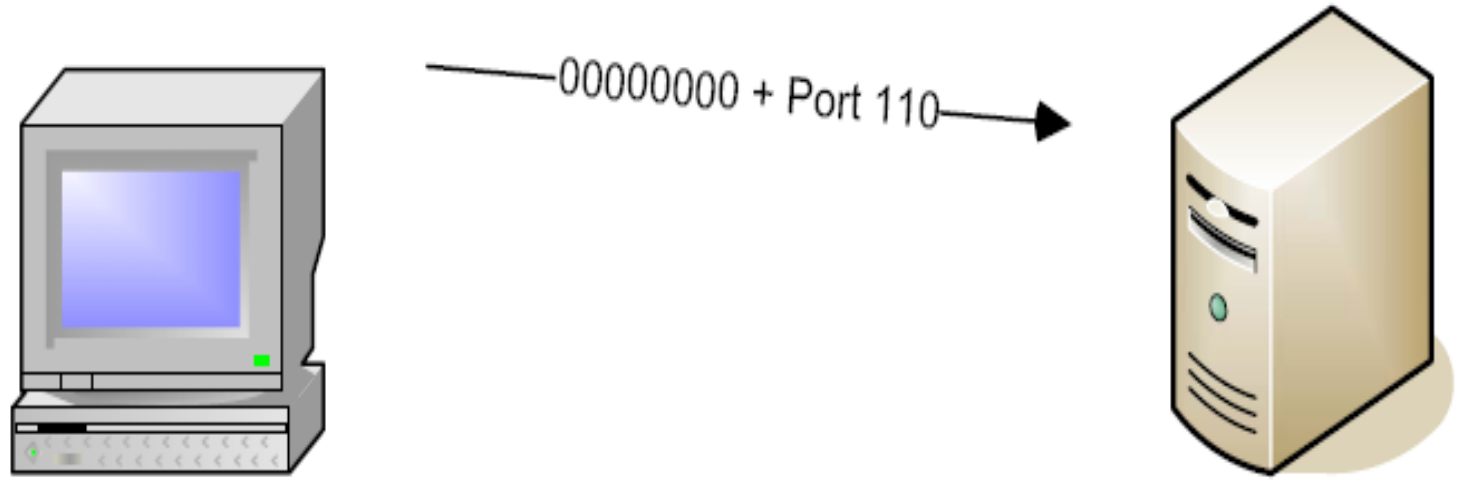
The null scan turns off all flags, creating a lack of TCP flags that should never occur in the real world.

If the port is closed, a RST frame should be returned:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.7]   TCP: D=438 S=36860     WIN=4096
[192.168.0.7]   [192.168.0.8]   TCP: D=36860 S=438 RST ACK=2135565682 WIN=0
```

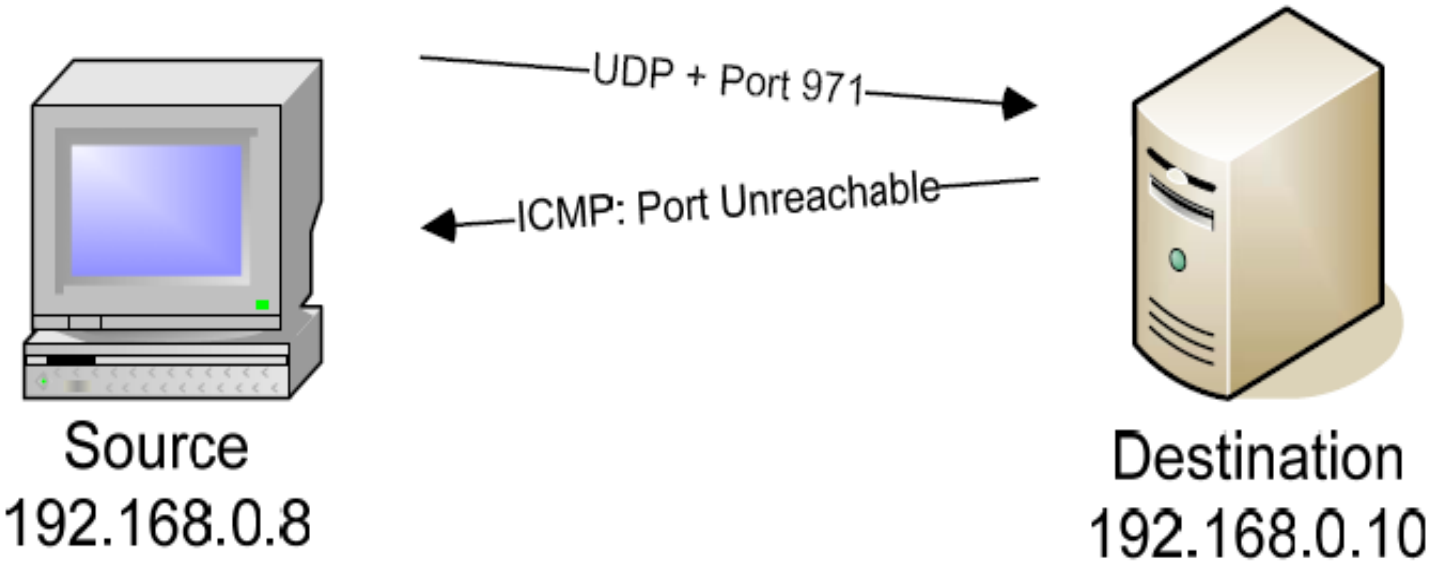
As expected, the response of a null scan to an open port results in no response:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.7]   TCP: D=110 S=36860      WIN=1024
```

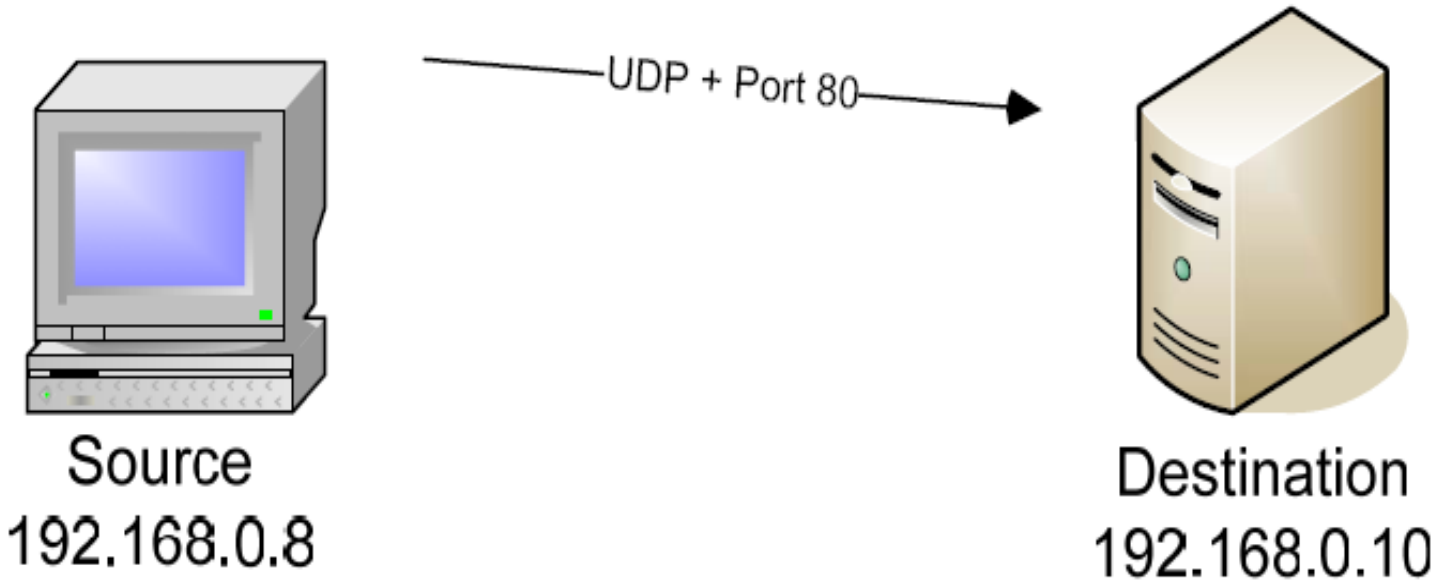
UDP Scan Operation

A station that responds with an ICMP port unreachable is clearly advertising a closed port:



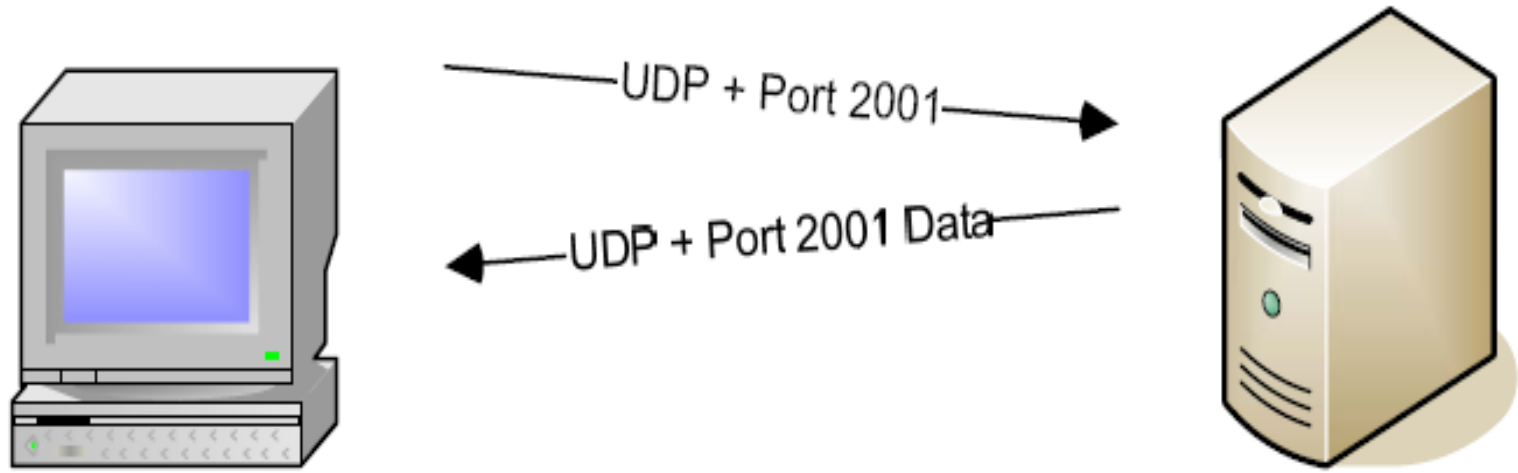

```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.10] UDP: D=971 S=43347 LEN=8
[192.168.0.10] [192.168.0.8]   ICMP: Destination unreachable (Port unreachable)
```

A station that doesn't respond to the UDP scan is considered to be open|filtered:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8] [192.168.0.10] UDP: D=80 S=43347 LEN=8
```

A station that responds with UDP data is indicative of an open port.




```
Source           Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]    [192.168.0.10] UDP: D=2001 S=43347 LEN=8
[192.168.0.10]   [192.168.0.8]  UDP: D=43347 S=2001 LEN=40
```

The nmap output shows the results of the UDP scan:

```
# nmap -sU -v 192.168.0.10

Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2005-04-11 12:44 EDT
Initiating UDP Scan against 192.168.0.10 [1478 ports] at 12:44
Discovered open port 2001/udp on 192.168.0.10
The UDP Scan took 1.47s to scan 1478 total ports.
Host 192.168.0.10 appears to be up ... good.
Interesting ports on 192.168.0.10:
(The 1468 ports scanned but not shown below are in state: closed)
PORT     STATE          SERVICE
123/udp  open|filtered ntp
137/udp  open|filtered netbios-ns
138/udp  open|filtered netbios-dgm
445/udp  open|filtered microsoft-ds
500/udp  open|filtered isakmp
1031/udp open|filtered iad2
1032/udp open|filtered iad3
1900/udp open|filtered UPnP
2001/udp open          wizard
4500/udp open|filtered sae-urn
MAC Address: 00:30:48:11:AB:5A (Supermicro Computer)

Nmap finished: 1 IP address (1 host up) scanned in 2.241 seconds
               Raw packets sent: 1489 (41.7KB) | Rcvd: 1470 (82.3KB)
#
```

IP Protocol Scan (`-sO`)

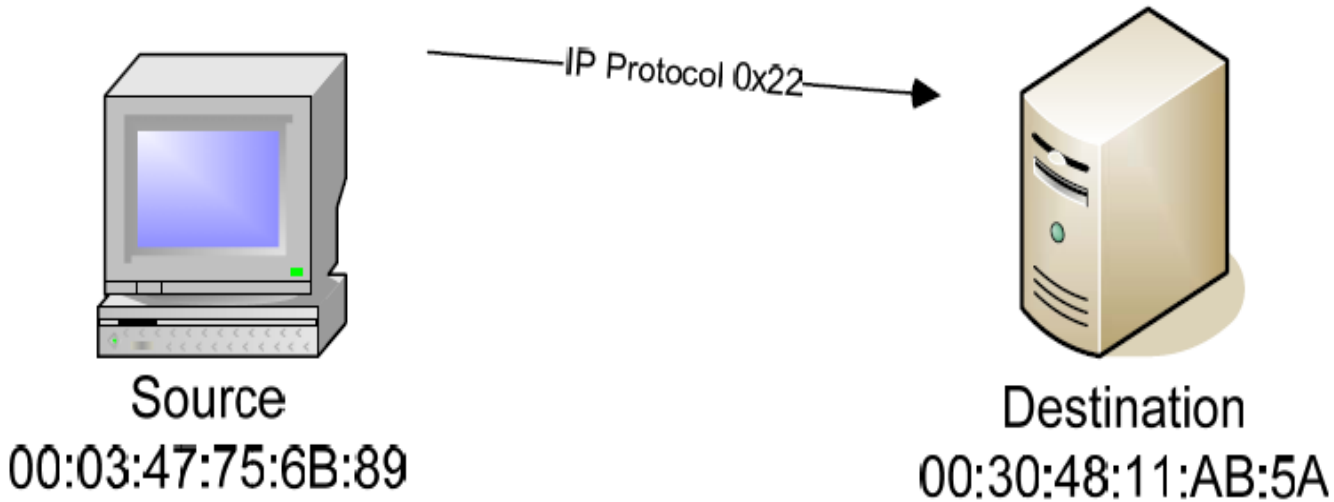
Requires Privileged Access: YES
Identifies TCP Ports: NO
Identifies UDP Ports: NO

The IP protocol scan is a bit different than the other nmap scans. The IP protocol scan is searching for additional IP protocols in use by the remote station, such as ICMP, TCP, and UDP. If a router is scanned, additional IP protocols such as EGP or IGP may be identified.

The list of IP protocols is found in the `nmap-protocols` file. If the `nmap-protocols` file isn't found, nmap reverts to the `/etc/protocols` file.

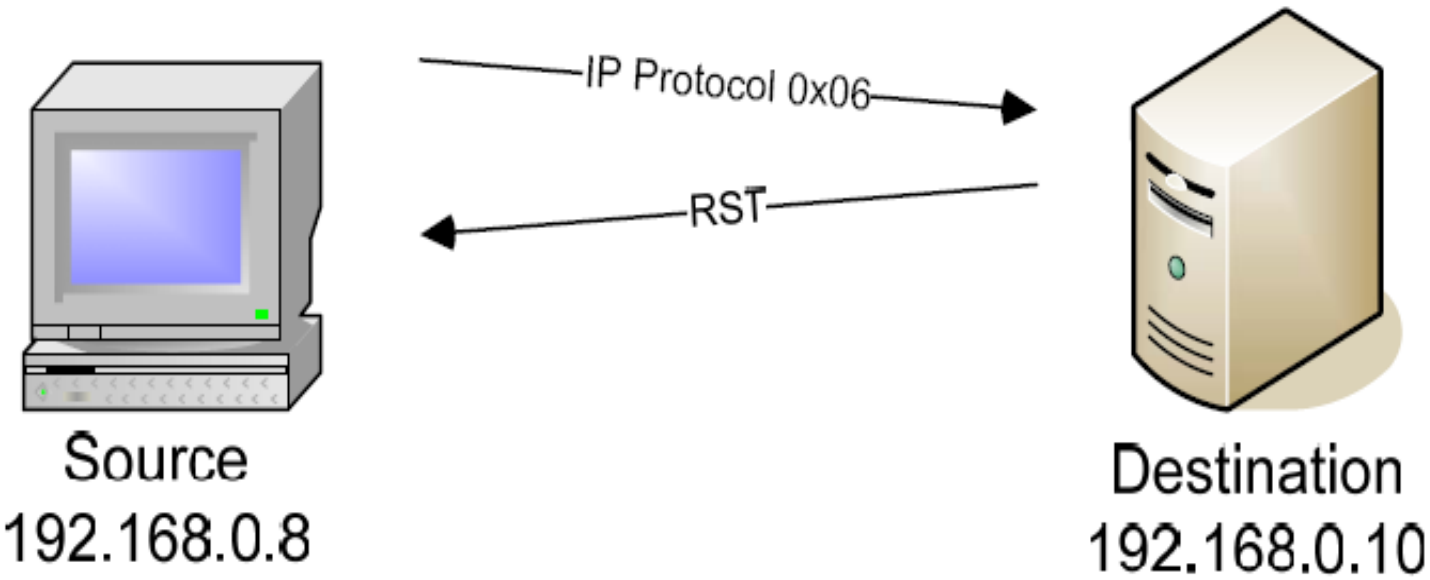
IP Protocol Scan Operation

An unavailable IP protocol does not respond to the scan. The MAC addresses are displayed to emphasize the IP layer conversation that occurs between the stations:




```
Source        Destination    Summary
-------------------------------------------------------------------------------
Intel 756B89 SprMcr11AB5A  IP: D=[192.168.0.10] S=[192.168.0.8] [0x22: xns-idp]
```

An available IP Protocol provides a response specific to the protocol type:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.10] TCP: D=44860 S=44860 ACK=637252255 WIN=1024
[192.168.0.10] [192.168.0.8]  TCP: D=44860 S=44860 RST WIN=0
```

The nmap output shows the IP protocol types available on a Windows-based workstation:

```
# nmap -sO -v 192.168.0.10

Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2005-04-11 12:46 EDT
Initiating IPProto Scan against 192.168.0.10 [256 ports] at 12:46
Discovered open port 6/ip on 192.168.0.10
Discovered open port 1/ip on 192.168.0.10
The IPProto Scan took 5.70s to scan 256 total ports.
Host 192.168.0.10 appears to be up ... good.
Interesting protocols on 192.168.0.10:
(The 253 protocols scanned but not shown below are in state: open|filtered)
PROTOCOL STATE    SERVICE
1        open     icmp
6        open     tcp
17       filtered udp
MAC Address: 00:30:48:11:AB:5A (Supermicro Computer)

Nmap finished: 1 IP address (1 host up) scanned in 6.620 seconds
               Raw packets sent: 511 (10.3KB) | Rcvd: 4 (194B)
#
```

ACK Scan (`-sA`)

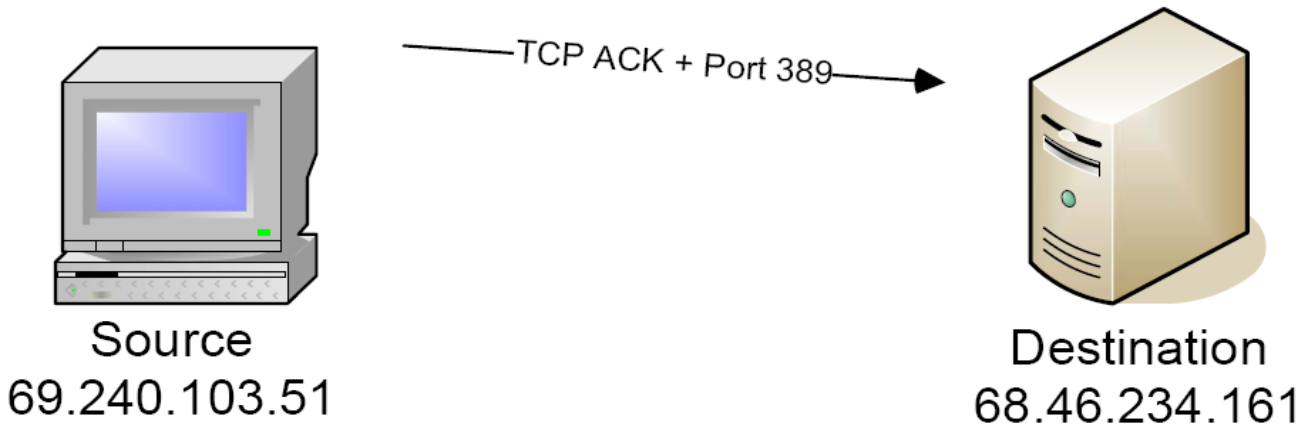
Requires Privileged Access: YES
Identifies TCP Ports: YES
Identifies UDP Ports: NO

Nmap's unique ACK scan will never locate an open port. The ACK scan only provides a "filtered" or "unfiltered" disposition because it never connects to an application to confirm an "open" state. At face value this appears to be rather limiting, but in reality the ACK scan can characterize the ability of a packet to traverse firewalls or packet filtered links.

ACK Scan (-sA) Contd.

ACK Scan Operation

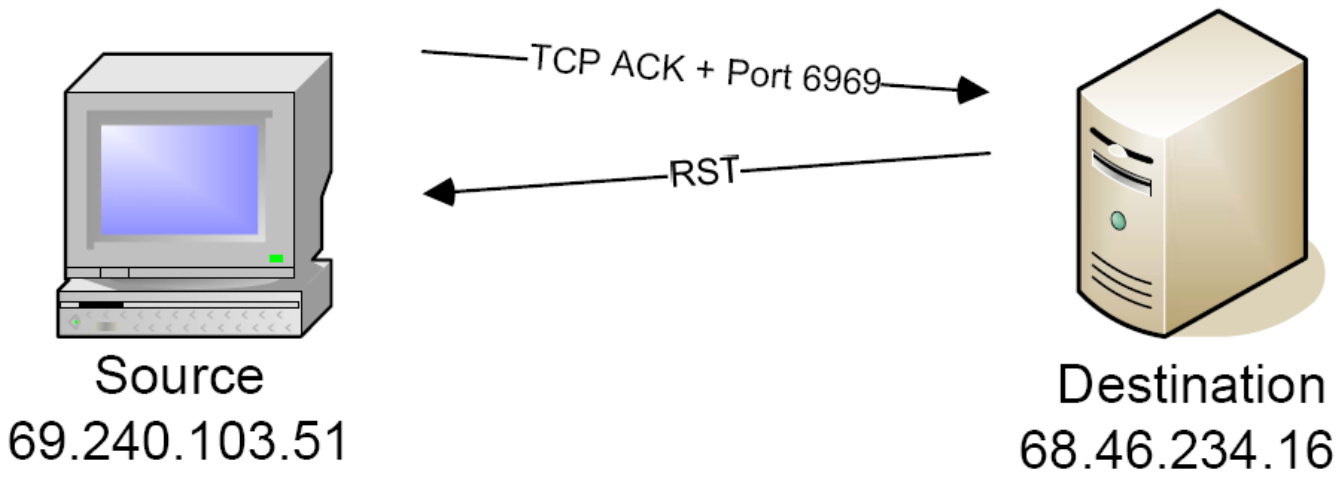
An ACK scan operates by sending a TCP ACK frame to a remote port. If there are no responses or an ICMP destination unreachable message is returned, then the port is considered to be "filtered:"




```
Source           Destination     Summary
-------------------------------------------------------------------------------
[69.240.103.51] [68.46.234.161] TCP: D=389 S=38667     ACK=0 WIN=3072
```

ACK Scan (-sA) contd.

If the remote port returns an RST packet, the connection between nmap and the remote device is categorized as unfiltered:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[69.240.103.51] [68.46.234.161] TCP: D=6969 S=38667     ACK=0 WIN=1024
[68.46.234.161] [69.240.103.51] TCP: D=38667 S=6969 RST WIN=0
```

Window Scan (-sW)

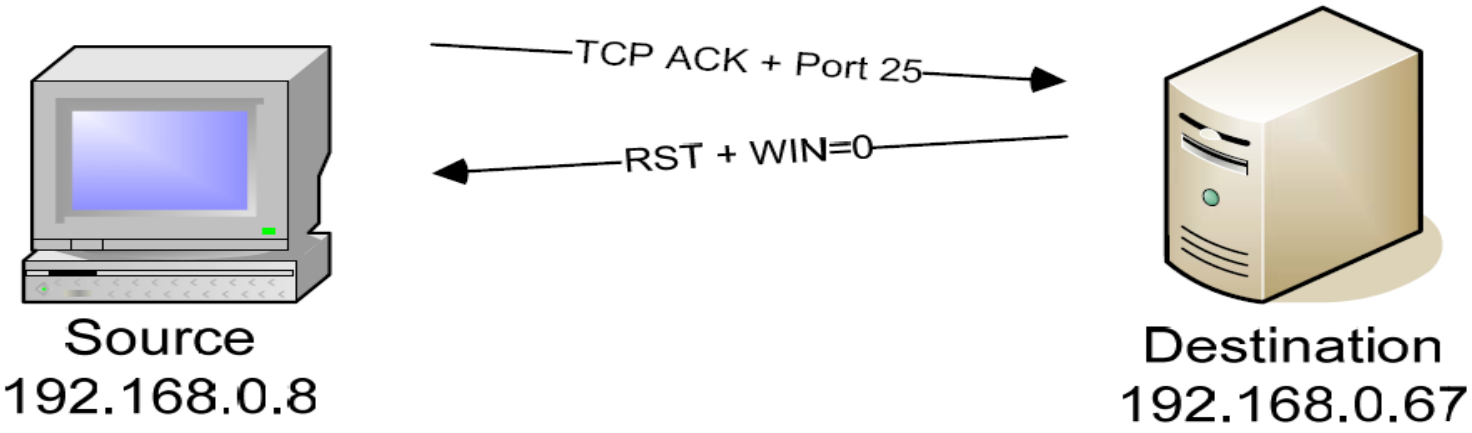
- Requires Privileged Access: YES
- Identifies TCP Ports: YES
- Identifies UDP Ports: NO

- The window scan is similar to an ACK scan, but the window scan has the advantage of identifying open ports.

Window Scan Operation

- The window scan is named after the TCP sliding window, not the operating system of a similar name.
- It's called the window scan because some TCP stacks have been found to provide specific window sizes when responding to an RST frame.

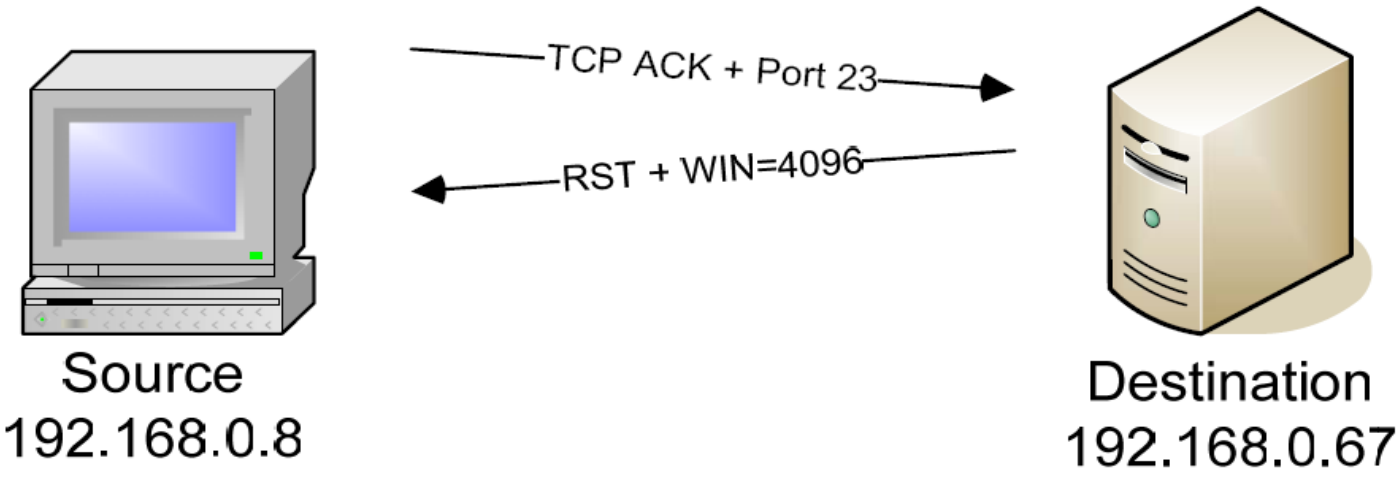
A RST frame response from a closed port responds with a window size of zero:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.67] TCP: D=25 S=62405      ACK=0 WIN=2048
[192.168.0.67]  [192.168.0.8]  TCP: D=62405 S=25 RST WIN=0
```

Window Scan Operation

When an open port is sent an ACK frame, the destination station still responds with a RST frame, but the window size is a non-zero value:




```
Source           Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]    [192.168.0.67]  TCP: D=23 S=62405      ACK=0 WIN=3072
[192.168.0.67]   [192.168.0.8]   TCP: D=62405 S=23 RST WIN=4096
```

The nmap output shows the results of the window scan:

```
# nmap -v -sW 192.168.0.67

Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2005-04-24 11:37 EDT
Initiating Window Scan against 192.168.0.67 [1663 ports] at 11:37
Discovered open port 23/tcp on 192.168.0.67
Discovered open port 21/tcp on 192.168.0.67
Discovered open port 111/tcp on 192.168.0.67
The Window Scan took 1.91s to scan 1663 total ports.
Host 192.168.0.67 appears to be up ... good.
Interesting ports on 192.168.0.67:
(The 1660 ports scanned but not shown below are in state: closed)
PORT    STATE SERVICE
21/tcp  open  ftp
23/tcp  open  telnet
111/tcp open  rpcbind
MAC Address: 00:10:A4:07:61:30 (Xircom)

Nmap finished: 1 IP address (1 host up) scanned in 2.749 seconds
               Raw packets sent: 1710 (68.4KB) | Rcvd: 1664 (76.5KB)
#
```

Ping Scan (-sP)

- Requires Privileged Access: NO
- Identifies TCP Ports: NO
- Identifies UDP Ports: NO

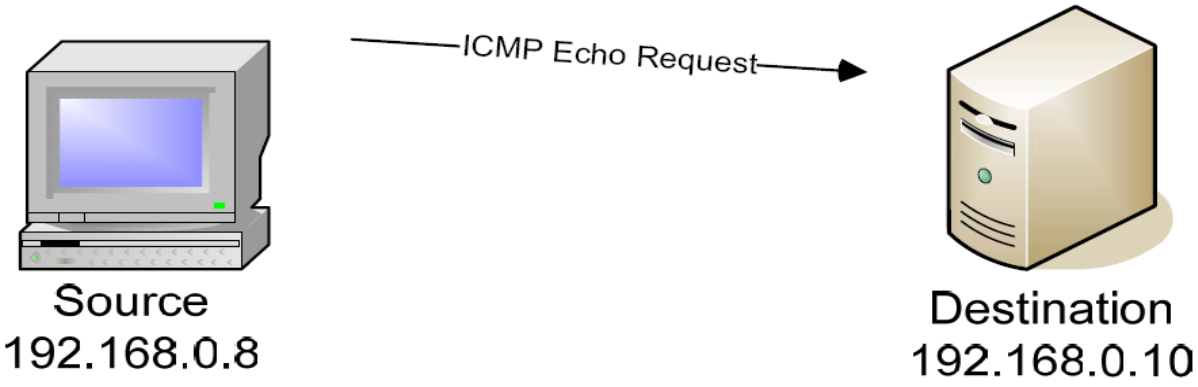
- The ping scan is one of the quickest scans that nmap performs, since no actual ports are queried.

- Unlike a port scan where thousands of packets are transferred between two stations, a ping scan requires only two frames. **This scan is useful for locating active devices or determining if ICMP is passing through a firewall.**

Ping Scan Operation

- The ping scan sends a single ICMP echo request from the nmap station to the destination device. A response from an active device will return an ICMP echo reply, unless the IP address is not available on the network or the ICMP protocol is filtered.

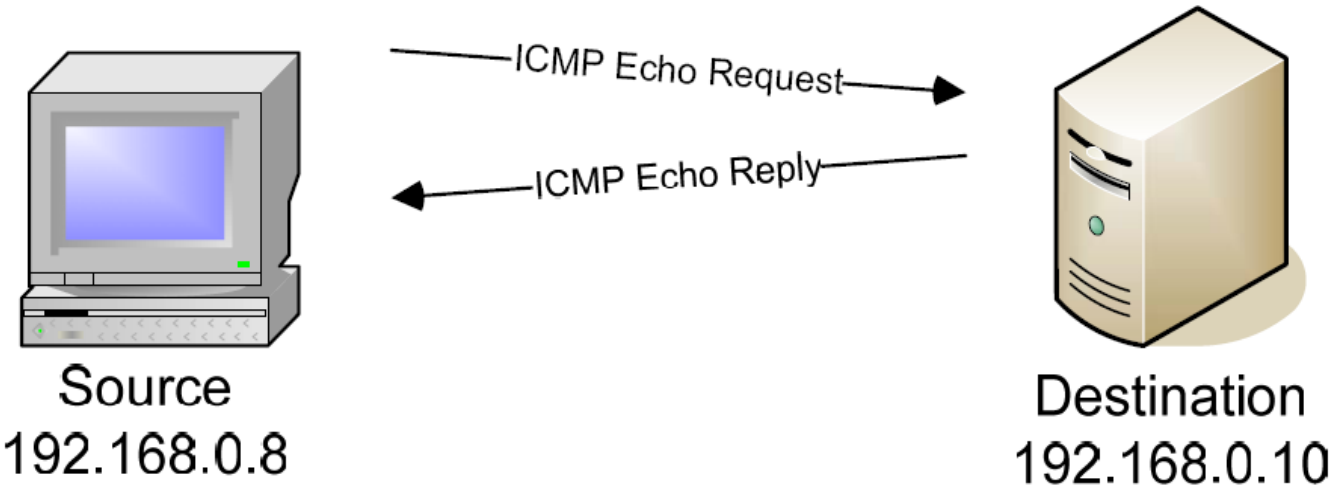
- If the station isn't available on the network or a packet filter is preventing ICMP packets from passing, there will be no response to the echo frame:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.10] ICMP: Echo
```

Ping Scan Operation Contd.

A response from an active host will return an ICMP echo reply, unless the IP address is not available on the network or ICMP is filtered.




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]   [192.168.0.10] ICMP: Echo
[192.168.0.10] [192.168.0.8]   ICMP: Echo reply
```

Version Detection (-sV)

- Privileged Access Required : NO
- Identifies TCP Ports : NO
- Identifies UDP Ports : NO

- Most of nmap's scanning methods are based around the identification of port numbers. However, the version detection scan is most interested in the software applications running on a remote device.

- For version detection to work properly, nmap relies on the nmap-service-probes file to provide a series of probes and the expected responses. If the nmap-service-probes support file is not available, the version detection scan will not run.

- Although other 3rd-party applications have implemented methods of version detection, the process used by the version detection scan is unique to nmap.

Version Detection Operation

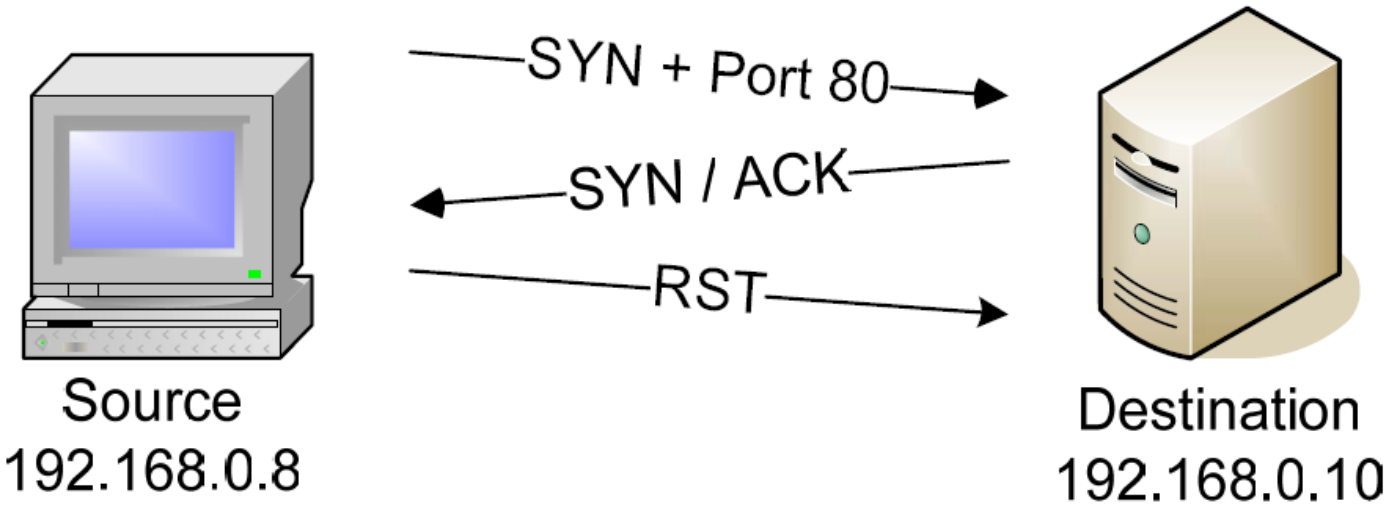
- The version detection scan runs in conjunction with another scan type that will identify open ports.

- If another scan type is not specified on the command line, nmap will run a TCP SYN scan (for a privileged user) or a TCP connect() scan (for non-privileged users) prior to running the version detection scan.

- If open ports are found, the version detection scan will begin the probing process with the remote device. The version detection scan communicates directly with the remote application to uncover as much information as possible.

Version Detection Operation Contd.

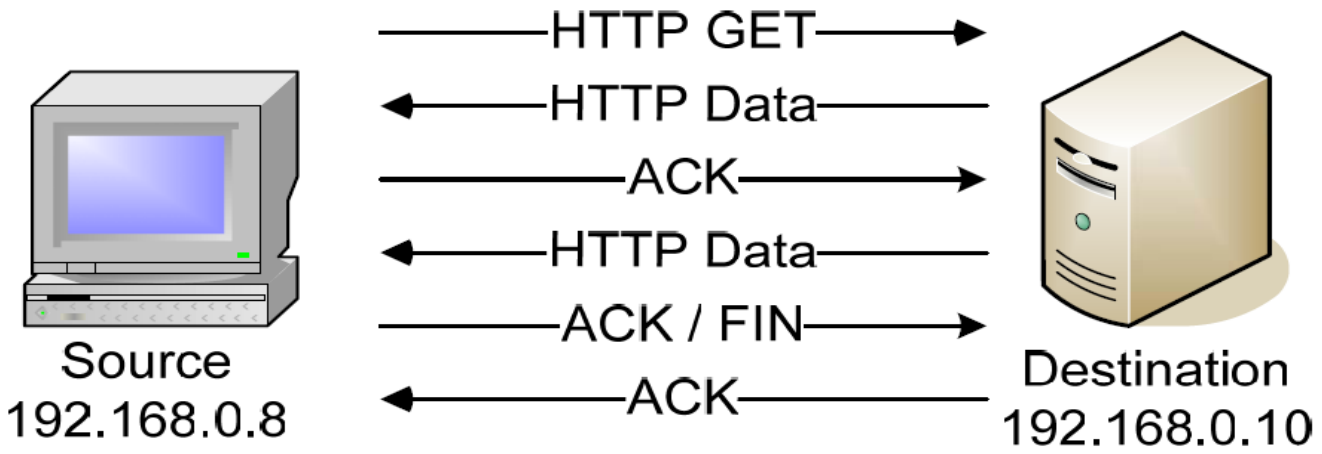
In this example, the default TCP SYN scan runs prior to the version detection scan to identify the open port:




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]  [192.168.0.10] TCP: D=80 S=54490 SYN SEQ=3420003014 LEN=0 WIN=1024
[192.168.0.10]  [192.168.0.8] TCP: D=54490 S=80 SYN ACK=3420003015 SEQ=1473373778 LEN=0 WIN=65535
[192.168.0.8]  [192.168.0.10] TCP: D=80 S=54490 RST WIN=0
```

Version Detection Operation Contd.

After the TCP SYN scan identifies port 80 as open, the version detection process begins. The process shown in this example is for this specific example. Other ports and application will operate differently.




```
Source          Destination     Summary
-------------------------------------------------------------------------------
[192.168.0.8]  [192.168.0.10]  TCP: D=80 S=39330 SYN SEQ=4090323855 LEN=0 WIN=5840
[192.168.0.10]  [192.168.0.8]  TCP: D=39330 S=80 SYN ACK=4090323856 SEQ=166204426 LEN=0 WIN=65535
[192.168.0.8]  [192.168.0.10]  TCP: D=80 S=39330 ACK=166204427 WIN<<2=5840
[192.168.0.8]  [192.168.0.10]  HTTP: C Port=39330 GET / HTTP/1.0
[192.168.0.10]  [192.168.0.8]  HTTP: R Port=39330 HTTP/1.1 Status=OK-1494 bytes of content \
[192.168.0.8]  [192.168.0.10]  TCP: D=80 S=39330 ACK=166205875 WIN<<2=8736
[192.168.0.10]  [192.168.0.8]  HTTP: Continuation of frame 2398;468 Bytes of data
[192.168.0.8]  [192.168.0.10]  TCP: D=80 S=39330 FIN ACK=166206344 SEQ=4090323874 LEN=0 WIN<<2=11632
[192.168.0.10]  [192.168.0.8]  TCP: D=39330 S=80 ACK=4090323875 WIN<<0=65517
```

Version Detection Operation Contd.

- The scan output displays the application information for each open port, although not all version numbers in this example were identified. The open ports were located by a TCP SYN scan that ran prior to the version scan.

- In this example, an open TCP port 520 was located by the SYN scan but the version scan did not recognize the service.

- A fingerprint was created and nmap provided an URL to use for submission of the unknown service (the URL has been intentionally obfuscated in this document for security reasons).

Version Detection Operation Contd.

- The version detection scan can include the --version_trace option, which provides a packet-by-packet display of the probing and fingerprinting process.

- This option is similar to the --packet_trace option, except the --version_trace option displays only a subset of the frames seen with the --packet_trace option.

- The version detection scan is one of the scans that runs automatically when the Additional, Advanced, and Aggressive scan (-A) is selected.

- The -A option provides an easy way to launch the version detection process in conjunction with other “discovery” scans.