**Using The nbtstat Utility**

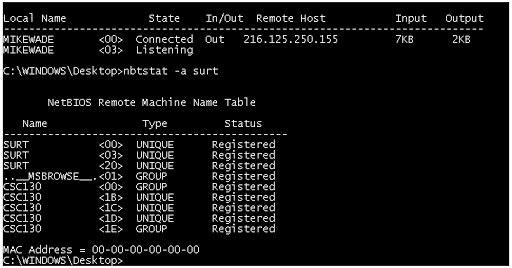
**The -a Switch**

Microsoft Windows uses an interface called **Network Basic Input/Output System (NetBIOS)**, which relates names with workstations and is an upper-layer interface that requires a transport protocol—usually TCP/IP. But IPv6 can be used as well. Deploying the **nbtstat** utility will achieve these three important things:

1. Track NetBIOS over TCP/IP statistics
2. Show the details of incoming and outgoing NetBIOS over TCP/IP connections
3. Resolve NetBIOS names

Understand that because NetBIOS name resolution is primarily a Windows network utility, the nbtstat command is available only in Windows-based operating systems.

Making use of the –a switch will get you a remote machine’s NetBIOS name table consisting of a list of every NetBIOS name the machine from which you’ve deployed the switch knows of. The –a switch produced the output from server surt shown below.

**Sample output of the nbtstat –a command**

So, using this switch arranges the NetBIOS name-table information in table form with output in four columns.

The **Name column** displays the NetBIOS name entry for the remote host machine. The next column gives you a unique two-digit hexadecimal identifier for the NetBIOS name. This identifier represents the last byte of the NetBIOS name depicted in the Name column, and it’s important because the same name could actually be used several times for the same machine. Plus, it identifies the specific service on the particular host that the name is referencing. The below tables list the hexadecimal identifiers for unique and group hostnames.

The **Type column** refers to the type of NetBIOS name being referenced. Unique NetBIOS names refer to individual hosts, and group names refer to logical groupings of workstations—either domains or workgroups.

The **Status column** gives you information about the status of a host’s NetBIOS even if it hasn’t been registered with the rest of the network.

| **Hex ID** | **Description** |
| --- | --- |
| 00 | General name for the computer. |
| 03 | Messenger service ID used to send messages between a WINS server and a workstation. This is the ID registered with a WINS server. |
| 06 | Remote Access Server (RAS) server service ID. |
| 21 | RAS client. |
| 53 | DNS. |
| 123 | Network Time Protocol (NTP). |
| 1B | Domain master browser ID. A NetBIOS name with this ID indicates the domain master browser. |
| 1F | Network Dynamic Data Exchange (NetDDE) service ID. |
| BE | Network monitor agent ID. |
| BF | Network monitor utility ID. |
| 01 | Master browser for a domain to other master browsers. |
| 20 | Internet group name ID. This ID is registered with the WINS server to indicate which computers are used for administrative purposes. |
| 1C | Domain group name ID. |
| 1D | Master browser name. |
| 1E | Normal group name. |

**The -A Switch**

The –A switch works just like the –a switch and will give you the same output, but the syntax of the command is different. Obviously, you use an uppercase A instead of a lowercase one, and you also have to include the host’s IP address instead of its NetBIOS name. To use it, type nbtstat followed by –A and finally the IP address of the specific host whose NetBIOS table you want to check out:

nbtstat –A 199.153.163.2

**The -c Switch**

Use the –c switch to display the local NetBIOS name cache on the workstation it’s running on. The below figure shows a sample output of the nbtstat –c command;

Boş data .. hatalı olabilir pre-class notu..

Each entry in this display shows the NetBIOS name, the hex ID for the service that was accessed, the type of NetBIOS name (unique or group), the IP address that the name resolves to, and its life. The *Life* value shows how many seconds each entry will live in the cache. When this time expires, the entry will be deleted.

**The -n Switch**

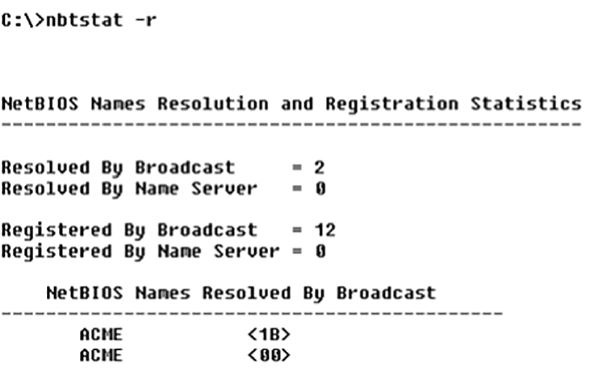
The –n switch will give you the local NetBIOS name table on a Windows device. The below figure shows an output that’s similar to the output of the –a switch except for one important thing: What you’re seeing is the NetBIOS name table for the machine you’re running the command on instead of that of another host.

**The -r Switch**

This switch is probably the one you’ll use most often when you want to get a hold of NetBIOS over TCP/IP (NBT) statistics because it tells you exactly how many NetBIOS names have been resolved to TCP/IP addresses. The below figure shows the sample output of the nbtstat –r command.

What you can see here is that the statistics are divided into two categories. First, there are the NetBIOS names resolution and registration statistics. This is how many names have been resolved or registered either by broadcasts on the local segment or via lookup from a WINS name server.

Next, you have the NetBIOS unique and group names and their associated hex IDs that were resolved or registered. In the figure, you can see that there’s a distinct lack of information regarding names resolved by a name server. What this means is that the output is telling you that there’s no WINS server operating—instead, all NetBIOS names were resolved by broadcast only.



**The -R Switch**

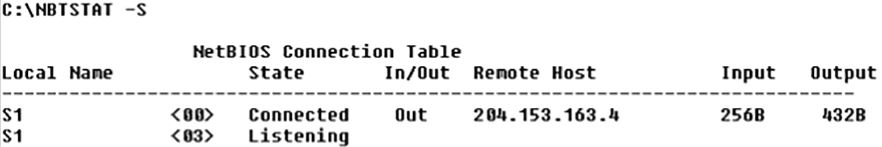
Unlike the –a and –A switches, -r and -R use the same letter but do not have anything in common.

Here’s an example. Let’s say you have a bad name in the NetBIOS name cache but the right name is in the LMHOSTS file instead. (The LMHOSTS file contains NetBIOS names of stations and their associated IP addresses.) Because the cache is consulted before the LMHOSTS file is, that bad address will remain in the cache until it expires. This command is used when you want to purge the NetBIOS name table cache and reload the LMHOSTS file into memory. You do that using the nbtstat command with the –R switch, like so:

nbtstat –R

**The -S Switch**

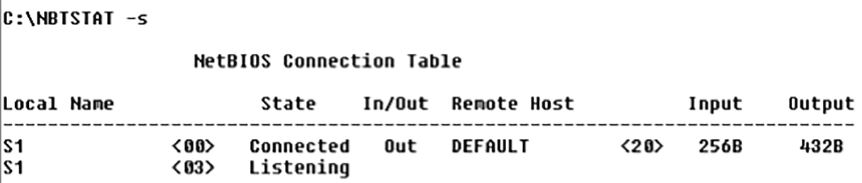
Using the -S switch will display the NetBIOS sessions table that lists all NetBIOS sessions to and from the host from which you issued the command. The -S switch displays both workstation and server sessions but lists remote addresses by IP address only. The below figure shows the sample output of the nbtstat -S command.



Here you can see the NetBIOS name is displayed along with its hex ID and the status of each session. An entry in the In/Out column determines whether the connection has been initiated from the computer on which you’re running nbtstat (outbound) or whether another computer has initiated the connection (inbound). The numbers in the Input and Output columns indicate in bytes the amount of data transferred between the stations.

**The -s Switch**

As with the -A and -a switches, the lowercase -s switch is similar to its uppercase sibling. The nbtstat -s command produces the same output as nbtstat -S except that it will also attempt to resolve remote-host IP addresses into hostnames. The below figure shows the sample output from the nbtstat -s command.



**The netstat Utility**

Using netstat is a great way to check out the inbound and outbound TCP/IP connections on your machine. You can also use it to view packet statistics like how many packets have been sent and received, the number of errors, and so on. When used without any options, netstat produces output similar to the following, which shows all the outbound TCP/IP connections. This utility is a great tool to use to determine the status of outbound web connections. Take a look:

C:\Users\clarusway>netstat

Active Connections

Proto Local Address Foreign Address State

TCP 192.168.1.22:49812 ec2-35-157-203-133:https ESTABLISHED

TCP 192.168.1.22:49824 ed-in-f188:5228 ESTABLISHED

TCP 192.168.1.22:50322 server-99-86-243-78:https ESTABLISHED

TCP 192.168.1.22:50918 54.239.31.91:https ESTABLISHED

TCP 192.168.1.22:51180 aeab55d76dd13c9bb:https ESTABLISHED

TCP 192.168.1.22:51211 ec2-18-205-93-210:https ESTABLISHED

TCP 192.168.1.22:51212 ec2-52-202-62-236:https CLOSE\_WAIT

TCP 192.168.1.22:51213 ec2-18-205-93-141:https CLOSE\_WAIT

TCP 192.168.1.22:51214 ec2-18-205-93-141:https CLOSE\_WAIT

TCP 192.168.1.22:51215 ec2-18-205-93-141:https CLOSE\_WAIT

TCP 192.168.1.22:51216 ec2-18-205-93-141:https CLOSE\_WAIT

TCP 192.168.1.22:51281 aeab55d76dd13c9bb:https ESTABLISHED

TCP 192.168.1.22:51318 52.46.68.59:https ESTABLISHED

TCP 192.168.1.22:51346 ec2-3-225-75-90:https ESTABLISHED

TCP 192.168.1.22:51377 52.114.128.43:https ESTABLISHED

TCP 192.168.1.22:51391 aeab55d76dd13c9bb:https ESTABLISHED

TCP 192.168.1.22:61298 ec2-52-202-62-228:https ESTABLISHED

TCP 192.168.1.22:61317 ec2-3-120-198-117:https ESTABLISHED

TCP 192.168.1.22:61320 ec2-3-120-198-117:https ESTABLISHED

TCP 192.168.1.22:61330 ec2-3-120-198-117:https ESTABLISHED

TCP 192.168.1.22:62010 51.105.249.228:https ESTABLISHED

The **Proto column** lists the protocol being used. The **Local Address** column lists the source address and the source port (source socket). The **Foreign Address** column lists the address of the destination machine (the hostname if it’s been resolved). If the destination port is known, it will show up as a well-known port. The **State column** indicates the status of each connection. This column shows statistics only for TCP connections because the *User Datagram Protocol (UDP)* establishes no virtual circuit to the remote device. Usually, this column indicates **ESTABLISHED** when a TCP connection between your computer and the destination computer has been established.

**💡Tip:**

* If the address of either your computer or the destination computer can be found in the HOSTS file on your computer, the destination computer’s name, rather than the IP address, will show up in either the Local Address or Foreign Address column.

The output of the netstat utility depends on the switch. By using the netstat /? command, we can see the options available to us.

C:\Users\clarusway>netstat /?

Displays protocol statistics and current TCP/IP network connections.

NETSTAT [-a] [-b] [-e] [-f] [-n] [-o] [-p proto] [-r] [-s] [-x] [-t] [interval]

-a Displays all connections and listening ports.

-b Displays the executable involved in creating each connection or

listening port. In some cases well-known executables host

multiple independent components, and in these cases the

sequence of components involved in creating the connection

or listening port is displayed. In this case the executable

name is in [] at the bottom, on top is the component it called,

and so forth until TCP/IP was reached. Note that this option

can be time-consuming and will fail unless you have sufficient

permissions.

-e Displays Ethernet statistics. This may be combined with the -s

option.

-f Displays Fully Qualified Domain Names (FQDN) for foreign

addresses.

-n Displays addresses and port numbers in numerical form.

-o Displays the owning process ID associated with each connection.

-p proto Shows connections for the protocol specified by proto; proto

may be any of: TCP, UDP, TCPv6, or UDPv6. If used with the -s

option to display per-protocol statistics, proto may be any of:

IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, or UDPv6.

-q Displays all connections, listening ports, and bound

nonlistening TCP ports. Bound nonlistening ports may or may not

be associated with an active connection.

-r Displays the routing table.

-s Displays per-protocol statistics. By default, statistics are

shown for IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, and UDPv6;

the -p option may be used to specify a subset of the default.

-t Displays the current connection offload state.

-x Displays NetworkDirect connections, listeners, and shared

endpoints.

-y Displays the TCP connection template for all connections.

Cannot be combined with the other options.

interval Redisplays selected statistics, pausing interval seconds

between each display. Press CTRL+C to stop redisplaying

statistics. If omitted, netstat will print the current

configuration information once.

**The -a Switch**

When you use the -a switch, the netstat utility displays all TCP/IP connections and all UDP connections. Below you can see the output produced by the netstat -a command.

C:\Users\clarusway>netstat -a

Active Connections

Proto Local Address Foreign Address State

TCP 192.168.1.22:49812 ec2-35-157-203-133:https ESTABLISHED

TCP 192.168.1.22:49824 ed-in-f188:5228 ESTABLISHED

TCP 192.168.1.22:50322 server-99-86-243-78:https ESTABLISHED

TCP 192.168.1.22:51211 ec2-18-205-93-210:https ESTABLISHED

TCP 192.168.1.22:51212 ec2-52-202-62-236:https CLOSE\_WAIT

TCP 192.168.1.22:51213 ec2-18-205-93-141:https CLOSE\_WAIT

TCP 192.168.1.22:51214 ec2-18-205-93-141:https CLOSE\_WAIT

TCP 192.168.1.22:51215 ec2-18-205-93-141:https CLOSE\_WAIT

TCP 192.168.1.22:51216 ec2-18-205-93-141:https CLOSE\_WAIT

TCP 192.168.1.22:51518 ec2-54-236-84-111:https ESTABLISHED

TCP 192.168.1.22:51548 185.11.14.41:http TIME\_WAIT

TCP 192.168.1.22:51549 185.11.14.41:http TIME\_WAIT

TCP 192.168.1.22:51550 185.11.14.41:http TIME\_WAIT

TCP 192.168.1.22:51563 99.86.243.5:https ESTABLISHED

TCP 192.168.1.22:51564 ec2-3-225-75-90:https ESTABLISHED

TCP 192.168.1.22:51579 52.114.132.73:https ESTABLISHED

TCP 192.168.1.22:51585 aeab55d76dd13c9bb:https ESTABLISHED

TCP 192.168.1.22:51597 server-99-86-245-89:https ESTABLISHED

TCP 192.168.1.22:61298 ec2-52-202-62-228:https ESTABLISHED

TCP 192.168.1.22:61317 ec2-3-120-198-117:https ESTABLISHED

TCP 192.168.1.22:61320 ec2-3-120-198-117:https ESTABLISHED

TCP 192.168.1.22:61330 ec2-3-120-198-117:https ESTABLISHED

TCP 192.168.1.22:62010 51.105.249.228:https ESTABLISHED

UDP [fe80::19ac:8efb:2c6e:f512%10]:1900 \*:\*

UDP [fe80::19ac:8efb:2c6e:f512%10]:2177 \*:\*

UDP [fe80::19ac:8efb:2c6e:f512%10]:58133 \*:\*

You can tell that UDP connections in the output are broadcasts because the destination address is listed as \* : \* (meaning “any address, any port”).

The most common use for the -a switch is to check the status of a TCP/IP connection that appears to be hung. You can determine if the connection is simply busy or is actually hung and no longer responding.

**💡Tip:**

* The **State** column in the figure has no entry for the UDP rows because UDP is not a connection-oriented protocol and, therefore, has no connection state.

**The -e Switch**

The -e switch displays a summary of all the packets that have been sent over the Network Interface Card (NIC) as of that instant. The Received and Sent columns show packets coming in as well as being sent:

C:\Users\clarusway>netstat -e

Interface Statistics

**Received** **Sent**

Bytes 652308520 724669536

Unicast packets 7476729 5597781

Non-unicast packets 6906 240780

Discards 0 0

Errors 0 1

Unknown protocols 0

You can use the -e switch to display the following categories of statistics:

* **Bytes** - The number of bytes transmitted or received since the computer was turned on. This statistic is useful for finding out if data is actually being transmitted and received or if the network interface isn’t doing anything at all.
* **Unicast Packets** - The number of packets sent from or received at this computer. To register in one of these columns, the packet must be addressed directly from one computer to another and the computer’s address must be in either the source or destination address section of the packet.
* **Non-unicast Packets** - The number of packets that weren’t directly sent from one workstation to another. For example, a broadcast packet is a non-unicast packet. The number of non-unicast packets should be smaller than the number of unicast packets. If the number of non-unicast packets is as high as or higher than that of unicast packets, too many broadcast packets are being sent over your network. Definitely find the source of these packets and make any necessary adjustments to optimize performance.
* **Discards** - The number of packets that were discarded by the NIC during either transmission or reception because they weren’t assembled correctly.
* **Errors** - The number of errors that occurred during transmission or reception. (These numbers may indicate problems with the network card.)
* **Unknown Protocols** - The number of received packets that the Windows networking stack couldn’t interpret. This statistic only shows up in the Received column because if the computer sent them, they wouldn’t be unknown.

Unfortunately, statistics don’t mean much unless they can be colored with time information. For example, if the Errors row shows 1 error, is that a problem? It might be if the computer has been on for only a few minutes. Unfortunately, the netstat utility doesn’t have a way of indicating how much time has elapsed for these statistics.

**The -r Switch**

You use the -r switch to display the current route table for a workstation so that you can see exactly how TCP/IP information is being routed.

C:\Users\clarusway>netstat –r

===========================================================================

Interface List

14...9c 5c 8e ce d9 c9 ......Intel(R) I211 Gigabit Network Connection

18...9c 5c 8e ce d9 ca ......Intel(R) Ethernet Connection (2) I219-V

15...76 c6 3b 00 62 86 ......Microsoft Wi-Fi Direct Virtual Adapter

8...76 c6 3b 00 6a 86 ......Microsoft Wi-Fi Direct Virtual Adapter #2

10...74 c6 3b 00 62 86 ......Broadcom 802.11ac Network Adapter

1...........................Software Loopback Interface 1

17...00 00 00 00 00 00 00 e0 Microsoft Teredo Tunneling Adapter

===========================================================================

IPv4 Route Table

===========================================================================

Active Routes:

Network Destination Netmask Gateway Interface Metric

0.0.0.0 0.0.0.0 192.168.1.1 192.168.1.22 35

127.0.0.0 255.0.0.0 On-link 127.0.0.1 331

127.0.0.1 255.255.255.255 On-link 127.0.0.1 331

127.255.255.255 255.255.255.255 On-link 127.0.0.1 331

192.168.1.0 255.255.255.0 On-link 192.168.1.22 291

192.168.1.22 255.255.255.255 On-link 192.168.1.22 291

192.168.1.255 255.255.255.255 On-link 192.168.1.22 291

224.0.0.0 240.0.0.0 On-link 127.0.0.1 331

224.0.0.0 240.0.0.0 On-link 192.168.1.22 291

255.255.255.255 255.255.255.255 On-link 127.0.0.1 331

255.255.255.255 255.255.255.255 On-link 192.168.1.22 291

===========================================================================

Persistent Routes:

None

IPv6 Route Table

===========================================================================

Active Routes:

If Metric Network Destination Gateway

17 331 ::/0 On-link

1 331 ::1/128 On-link

17 331 2001::/32 On-link

17 331 2001:0:2851:782c:148e:f3fd:6aff:55b8/128

On-link

10 291 fe80::/64 On-link

17 331 fe80::/64 On-link

17 331 fe80::148e:f3fd:6aff:55b8/128

On-link

10 291 fe80::19ac:8efb:2c6e:f512/128

On-link

1 331 ff00::/8 On-link

10 291 ff00::/8 On-link

17 331 ff00::/8 On-link

===========================================================================

Persistent Routes:

None

**The -s Switch**

Using the -s switch displays a variety of TCP, UDP, IP, and ICMP protocol statistics. But be warned—the output you’ll get is really long, which may or may not be okay for you.

C:\Users\clarusway>netstat -s

IPv4 Statistics

Packets Received = 85199526

Received Header Errors = 0

Received Address Errors = 113

Datagrams Forwarded = 0

Unknown Protocols Received = 49

Received Packets Discarded = 9859

Received Packets Delivered = 85614599

Output Requests = 60765459

Routing Discards = 0

Discarded Output Packets = 5954

Output Packet No Route = 202

Reassembly Required = 10

Reassembly Successful = 4

Reassembly Failures = 0

Datagrams Successfully Fragmented = 0

Datagrams Failing Fragmentation = 0

Fragments Created = 0

IPv6 Statistics

Packets Received = 261062

Received Header Errors = 0

Received Address Errors = 321

Datagrams Forwarded = 0

Unknown Protocols Received = 0

Received Packets Discarded = 981

Received Packets Delivered = 263904

Output Requests = 244350

Routing Discards = 0

Discarded Output Packets = 539

Output Packet No Route = 0

Reassembly Required = 0

Reassembly Successful = 0

Reassembly Failures = 0

Datagrams Successfully Fragmented = 0

Datagrams Failing Fragmentation = 0

Fragments Created = 0

ICMPv4 Statistics

Received Sent

Messages 4104 8133

Errors 0 0

Destination Unreachable 2443 5570

Time Exceeded 437 0

Parameter Problems 0 0

Source Quenches 0 0

Redirects 0 0

Echo Replies 1224 0

Echos 0 2563

Timestamps 0 0

Timestamp Replies 0 0

Address Masks 0 0

Address Mask Replies 0 0

Router Solicitations 0 0

Router Advertisements 0 0

ICMPv6 Statistics

Received Sent

Messages 934 2128

Errors 0 0

Destination Unreachable 184 185

Packet Too Big 0 0

Time Exceeded 11 0

Parameter Problems 0 0

Echos 0 24

Echo Replies 0 0

MLD Queries 0 0

MLD Reports 0 0

MLD Dones 0 0

Router Solicitations 0 92

Router Advertisements 27 0

Neighbor Solicitations 360 1445

Neighbor Advertisements 376 382

Redirects 0 0

Router Renumberings 0 0

TCP Statistics for IPv4

Active Opens = 67905

Passive Opens = 1709

Failed Connection Attempts = 40609

Reset Connections = 5402

Current Connections = 43

Segments Received = 64039477

Segments Sent = 47069056

Segments Retransmitted = 162012

TCP Statistics for IPv6

Active Opens = 258

Passive Opens = 86

Failed Connection Attempts = 222

Reset Connections = 56

Current Connections = 0

Segments Received = 5728

Segments Sent = 4332

Segments Retransmitted = 717

UDP Statistics for IPv4

Datagrams Received = 22383699

No Ports = 6416

Receive Errors = 260328

Datagrams Sent = 14243567

UDP Statistics for IPv6

Datagrams Received = 267432

No Ports = 966

Receive Errors = 15

Datagrams Sent = 239189

**The -p Switch**

Like the -n switch, the -p switch is a modifier that’s usually used with the -s switch to specify which protocol statistics to list in the output (IP, TCP, UDP, or ICMP). For example, if you want to view only ICMP statistics, you use the -p switch like so:

netstat -s -p ICMP

The netstat utility then displays the ICMP statistics instead of the entire gamut of TCP/IP statistics that the -s switch will typically flood you with. For a different example, let’s use the -s and -p switches to retrieve some IPv6 information:

C:\Users\clarusway>netstat -s -p IPV6

IPv6 Statistics

Packets Received = 261062

Received Header Errors = 0

Received Address Errors = 321

Datagrams Forwarded = 0

Unknown Protocols Received = 0

Received Packets Discarded = 981

Received Packets Delivered = 263904

Output Requests = 244359

Routing Discards = 0

Discarded Output Packets = 539

Output Packet No Route = 0

Reassembly Required = 0

Reassembly Successful = 0

Reassembly Failures = 0

Datagrams Successfully Fragmented = 0

Datagrams Failing Fragmentation = 0

Fragments Created = 0

**The -n Switch**

The -n switch is a modifier for the other switches. When used with them, it reverses the natural tendency of netstat to use names instead of network addresses. In other words, when you use the -n switch, the output always displays network addresses instead of their associated network names. Following is output from the netstat command used with the netstat -n command. It’s showing the same information but with IP addresses instead of names:

C:\Users\clarusway>netstat -n

Active Connections

Proto Local Address Foreign Address State

TCP 192.168.1.22:49812 35.157.203.133:443 ESTABLISHED

TCP 192.168.1.22:49824 74.125.143.188:5228 ESTABLISHED

TCP 192.168.1.22:52352 18.205.93.208:443 ESTABLISHED

TCP 192.168.1.22:52354 18.205.93.141:443 CLOSE\_WAIT

TCP 192.168.1.22:52355 52.202.62.236:443 CLOSE\_WAIT

TCP 192.168.1.22:52356 18.205.93.141:443 CLOSE\_WAIT

TCP 192.168.1.22:52357 18.205.93.141:443 CLOSE\_WAIT

TCP 192.168.1.22:52358 18.205.93.141:443 CLOSE\_WAIT

TCP 192.168.1.22:52458 99.86.243.78:443 ESTABLISHED

TCP 192.168.1.22:52499 52.114.77.34:443 ESTABLISHED

TCP 192.168.1.22:52508 18.205.44.71:443 ESTABLISHED

TCP 192.168.1.22:52515 75.2.53.94:443 ESTABLISHED

TCP 192.168.1.22:52517 75.2.53.94:443 ESTABLISHED

TCP 192.168.1.22:52566 99.83.135.170:443 ESTABLISHED

TCP 192.168.1.22:52584 99.86.245.105:443 ESTABLISHED

TCP 192.168.1.22:52602 99.86.243.98:443 ESTABLISHED

TCP 192.168.1.22:52617 99.83.135.170:443 ESTABLISHED

TCP 192.168.1.22:52668 52.114.7.36:443 TIME\_WAIT

TCP 192.168.1.22:52679 3.225.75.90:443 ESTABLISHED

TCP 192.168.1.22:52688 99.83.135.170:443 ESTABLISHED

TCP 192.168.1.22:52705 52.155.169.137:443 TIME\_WAIT

TCP 192.168.1.22:61298 52.202.62.228:443 ESTABLISHED

TCP 192.168.1.22:61317 3.120.198.117:443 ESTABLISHED

TCP 192.168.1.22:61320 3.120.198.117:443 ESTABLISHED

TCP 192.168.1.22:61330 3.120.198.117:443 ESTABLISHED

TCP 192.168.1.22:62010 51.105.249.228:443 ESTABLISHED

**Using The tcpdump Utility**

**Using tcpdump**

The tcpdump utility is used to read either packets captured live from a network or packets that have been saved to a file. Although there is a Windows version called windump, tcpdump only works on Unix-like operating systems.

* Use this command to capture traffic on all interfaces:

# tcpdump -i any

* Here is the command to capture traffic on a particular interface:

# tcpdump -i eth0

* And to filter traffic by IP, whether it’s the source or the destination, use this command:

# tcpdump host 192.168.5.5

**Using The File Transfer Protocol**

**Using the File Transfer Protocol**

**File Transfer Protocol (FTP)** is a subset of TCP/IP and that FTP is used for the transfer of files. In recent years, FTP has become a truly cross-platform protocol for transferring files. Almost every client and server platform has implemented FTP. Windows is no exception. Its TCP/IP stack comes with a command-line ftp utility.

To start the ftp utility, enter ftp at a command prompt. The result is an ftp command prompt:

C:\Users\clarusway>ftp

ftp>

From this prompt, you can open a connection to an FTP server and upload and download files as well as change the way FTP operates. To display a list of all the commands you can use at the ftp command prompt, type help or ? and press Enter. To get help on a specific command, type help, a space, and then the name of the command. Here is some output from the help command:

C:\Users\clarusway>ftp

ftp> ?

Commands may be abbreviated. Commands are:

! delete literal prompt send

? debug ls put status

append dir mdelete pwd trace

ascii disconnect mdir quit type

bell get mget quote user

binary glob mkdir recv verbose

bye hash mls remotehelp

cd help mput rename

close lcd open rmdir

**💡Tip:**

* Third-party applications are available that provide a GUI interface for FTP, which is easier to use than a command line.

**Starting FTP and Logging In to an FTP Server**

Of the two FTP file operations (download and upload), the ability to download files is definitely the more crucial for you to have down as a network technician or sysadmin.

The first steps in starting an FTP download session are to determine the address of the FTP site and start the ftp utility. The FTP site typically has the same name as the website except that the first three characters are ftp instead of www. For example, Microsoft’s website is www.microsoft.com. Its FTP site, on the other hand, is ftp.microsoft.com.

First, start the ftp utility as demonstrated in the preceding section, and then follow these steps:

1. At the ftp command prompt, type open, a space, and the name of the FTP server, like this:

C:\Users\clarusway> ftp

ftp> open ftp.claruswaytrainer.com

Connected to ftp.claruswaytrainer.com.

220---------- Welcome to Pure-FTPd [TLS] ----------

220-You are user number 1 of 100 allowed.

220-Local time is now 11:45. Server port: 21.

220-IPv6 connections are also welcome on this server.

220 You will be disconnected after 15 minutes of inactivity.

User (ftp.claruswaytrainer.com:(none)): enter

230 Anonymous user logged in

ftp>

As shown here, if the FTP server is available and running, you’ll receive a response welcoming you to the server and asking you for a username. Right now, we used *Anonymous* as the username (enabled by default on the FTP server), which means that anyone can log in to it.

You can also start an FTP session by typing ftp, a space, and the address of the FTP server. This allows you to start the ftp utility and open a connection in one step. Here’s an example:

C:\Users\clarusway> ftp ftp.claruswaytrainer.com

1. Enter a valid username, and press Enter.
2. Enter your password, and press Enter.

If you enter the wrong username and/or password, the server will tell you so by displaying the following and leaving you at the ftp command prompt:

530 Login Incorrect

Login failed.

This means you’ve got to try again and must start the login process over. If you’re successful, the FTP server will welcome you and drop you back at the ftp command prompt.

**Downloading Files**

After you log in to the FTP server, you’ll navigate to the directory that contains the files you want. The FTP command-line interface is similar to the DOS command-line interface.

The below table lists and describes the common navigation commands for FTP. After you navigate to the directory and find the file you want to download, it’s time to set the parameters for the type of file. Files come in two types:

* ASCII, which contains text
* Binary, which is all other files

If you set ftp to the wrong type, the file you download will contain gibberish. So if you’re in doubt, set ftp to download files as binary files. Check out the below table.

| **Command** | **Description** |
| --- | --- |
| ls | Short for list. Displays a directory listing. Very similar to the DIR command in MS-DOS. |
| cd | Short for change directory. Works almost identically to the MS-DOS CD command. Use it to change to a different directory and navigate the server’s directory structure. |
| pwd | Short for print working directory. Displays the current directory on the server. Useful if you forget where you are when changing to several locations on the server. |
| lcd | Short for local change directory. Displays and changes the current directory on the local machine. Useful when you are downloading a file and aren’t in the directory where you want to put the file. |

To set the file type to ASCII, type ascii at the ftp command prompt. ftp will respond by telling you that the file type has been set to A (ASCII):

ftp>ascii

Type set to A

To set the file type to binary, type binary at the ftp command prompt. ftp will respond by telling you that the file type has been set to I (binary):

ftp>binary

Type set to I

To download the file, just use the get command like this:

ftp>get test.exe

200 PORT command successful.

150 Opening BINARY mode data connection for 'test.exe'

(567018 bytes).

The file will start downloading to your hard drive. Unfortunately, with its default settings, the ftp utility doesn’t give you any indication of the progress of the transfer. When the file has downloaded, the ftp utility will display the following message and return you to the ftp command prompt:

226 Transfer complete.

567018 bytes received in 116.27 seconds (4.88 Kbytes/sec)

**Uploading Files**

To upload a file to an FTP server, you’ve got to have rights on that specific server. These rights are assigned on a directory-by-directory basis. To upload a file, log in and then follow these steps:

1. At the ftp command prompt, type lcd to navigate to the directory on the local machine where the file resides.
2. Type cd to navigate to the destination directory.
3. Set the file type to ASCII or binary.
4. Use the put command to upload the file.

The syntax of the put command looks like this:

ftp> put local file destination file

Let’s say you want to upload a file called test.txt on the local server but you want it to be called my.txt on the destination server. To accomplish that, use the following command:

ftp> put test.txt my.txt

You’ll get the following response:

200 PORT command successful.

150 Opening BINARY mode data connection for my.txt

226 Transfer complete.

743622 bytes sent in 0.55 seconds (1352.04 Kbytes/sec)

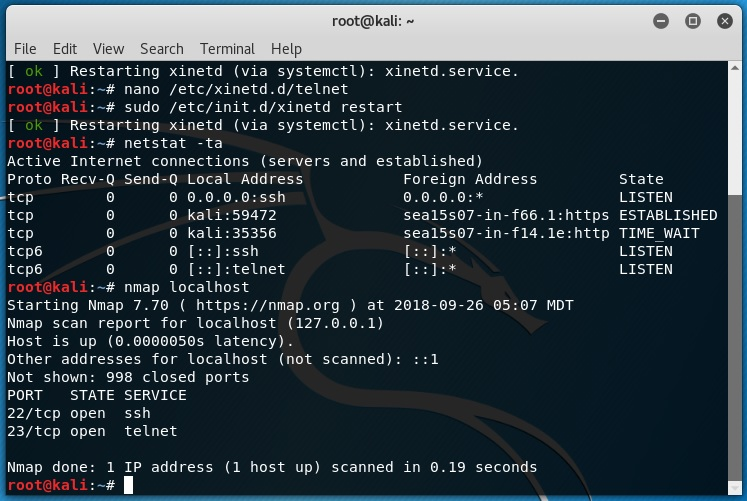
**💡Tips:**

* You can upload multiple files using the **mput** command. Simply type **mput**, a space, and then a wildcard that specifies the files. For example, to upload all the text files in a directory, type **mput \*.txt**. And in the same way, you can download multiple files with **mget** command.
* **get** and **put** commands require the perfect match of the file names intented to be downloaded and uploaded. On the other hand, **mget** and **mput** commands can run on partial match, a pattern with wildcards or asteriks.

**Using The Telnet Utility**

**Using the Telnet Utility**

Part of the TCP/IP protocol suite, **Telnet** is a virtual terminal protocol utility that allows you to make connections to remote devices, gather information, and run programs. Telnet was originally developed to open terminal sessions from remote Unix workstations to Unix servers. Although it’s still used for that purpose, we now use it as a troubleshooting tool as well. The below figure shows the basic Telnet interface as it’s being used to start a terminal session on a remote Unix host.



In today’s Windows environments, Telnet is a basic command-line tool for testing TCP connections. You can telnet to any TCP port to see if it’s responding—something that’s especially useful when checking *Simple Mail Transfer Protocol (SMTP)* and *HTTP* (web) ports.

**How to Enable Telnet in Windows**

Because most people have the Windows 10 operating system running on their PCs these days, it’s good to know that, by default, these operating systems install without Telnet available. But there’s a way around that one—if you really must have a Telnet client enabled in these operating systems, here’s how to do it:

1. Open Control Panel.
2. Select Programs And Features.
3. In the left column, select Turn Windows Features On Or Off
4. Select the Telnet checkbox (and any other obscure services you may want enabled), and wait while Windows installs for a while and then reboots.

Now you can go to Start and then type telnet in the Start search box to get a Telnet window to open for you. You can also open a DOS prompt and just type telnet from there. Here are the options that Windows provides with Telnet:

Microsoft Telnet> ?

Commands may be abbreviated. Supported commands are:

c - close close current connection

d - display display operating parameters

o - open hostname [port] connect to hostname (default port 23).

q - quit exit telnet

set - set set options (type 'set ?' for a list)

sen - send send strings to server

st - status print status information

u - unset unset options (type 'unset ?' for a list)

?/h - help print help information

**Don’t Use Telnet, Use Secure Shell**

Telnet is totally insecure because it sends all data in crystal-clear text—including your name and password. If Microsoft doesn’t even enable it on its latest OSs, then you know it really must be insecure.

So if you shouldn’t use Telnet, what should you use instead? **Secure Shell (SSH)** is your answer. It provides the same options as Telnet, plus a lot more; but most importantly, it doesn’t send any data in cleartext. The thing is, your servers, routers, and other devices need to be enabled with SSH, and it’s not configured by default on most devices.

Some configuration is usually necessary if you want things to work as they really should, and yes, sometimes it’s a little painful to get everything running smoothly, but it’s all worth it in the long run.

**Secure Shell (ssh)**

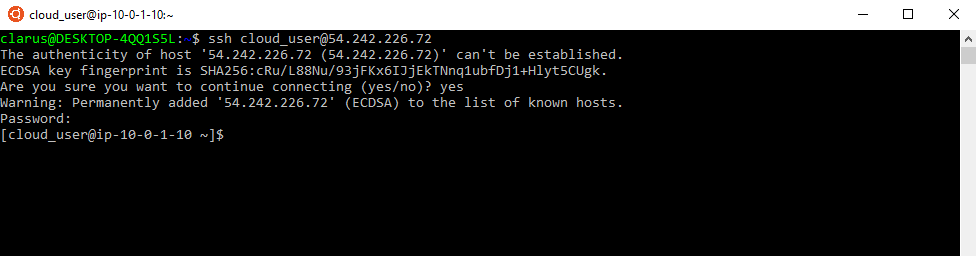
The **secure shell** or **ssh** is a collection of tools using a secure protocol for communications with remote computers.

It is a protocol used to securely connect to a remote server/system. ssh is secure in the sense that it transfers the data in encrypted form between the host and the client. It transfers inputs from the client to the host and relays back the output. ssh runs at TCP/IP port 22.

ssh user\_name@host(IP/Domain-name)

ssh command instructs the system to establish an encrypted secure connection with the host machine.

* user\_name represents the account that is being accessed on the host.
* host refers to the machine which can be a computer or a router that is being accessed. It can be an IP address (e.g. 54.164.151.235) or domain name(e.g. www.clarusway.com).



Q: How do you use **ssh** to connect to a **remote** server in Linux.  
A: Most servers in the world are run on Linux servers. They’re dependable, affordable and highly configurable. However, servers aren’t always accessed, nor accessible, directly. Hence they require remote access. The most frequently used, and secure, method of accessing servers remotely is via SSH, otherwise known as Secure Shell.  
For connect to remote server, first we should open to linux terminal and then type  
ssh user@www.remote\_server\_name.com  
or  
ssh user@82.178.72.19

 - Interview Q&A

**Using The scp and curl Utility**

**scp Command**

scp (Secure Copy) is a command-line tool that is used to transfer files and directories across the systems securely over the network. When we use scp command to copy files and directories from our local system to a remote system then in the backend it makes **ssh connection** to a remote system.

**Syntax**:

scp <options> <files\_or\_directories> user@target\_host:/<folder>

scp <options> user@target\_host:/files <folder\_local\_system>

Some of the most widely used options in scp command are listed below:

| **command** | **Explanation** |
| --- | --- |
| -C | Enable Compression |
| -i | identity File or private key |
| -l | limit the bandwidth while copying |
| -P | ssh port number of the target host |
| -r | Copy files and directories recursively |
| -p | Preserves modification times, access times, and modes from the original file |
| -q | Disables the progress meter |

Examples:

* Copies the file "test.txt" from a remote host to the localhost:

scp your\_username@hostname:text.txt /some/local/directory

* Copies the file "test.txt" from the local host to a remote host:

scp text.txt your\_username@hostname:/some/local/directory

* Copies multiple files from the remote host to your current directory on the localhost:

scp text.txt your\_username@hostname: /some/local/directory/\{a,b,c\}

Q: How to copy a file from a remote server to a local machine?  
A: If you are on the local computer wanting to receive file from a remote computer:  
**scp** username@remote:/file/to/send /where/to/put

 - Interview Q&A

**curl Command**

curl is a command-line tool to transfer data to or from a server, using any of the supported protocols.

**Syntax**:

curl [options] [URL...]

user@clarusway:~$ curl https://www.clarusway.com

* **-o**: Saves the downloaded file on the local machine with the name provided in the parameters.

curl -o [file\_name] [URL...]

Example:

user@clarusway:~$ curl -o hello.zip ftp://speedtest.tele2.net/1MB.zip

**Network Configuration Files**

**Network Configuration Files**

The graphical help tools use a few basic commands to edit a specific set of network configuration files. The exact names and location of the configuration files in the file system depend largely on your distribution and version of Linux.

| **File** | **Description** |
| --- | --- |
| /etc/resolv.conf | List DNS servers for internet domain name resolution. Manual page for: /etc/resolv.conf |
| /etc/hosts | Lists hosts to be resolved locally (not by DNS). Manual page for: /etc/hosts |
| /etc/nsswitch.conf | List order of host name search. Typically look at local files, then NIS server, then DNS server. Manual page for: /etc/nsswitch.conf |
| Red Hat/Fedora/CentOS: /etc/sysconfig/network | Specify network configuration. eg. Static IP, DHCP, NIS, etc. |
| Red Hat/Fedora/CentOS: /etc/sysconfig/network-scripts/ifcfg-device | Specify TCP network information. |
| Ubuntu/Debian: /etc/network/interfaces | Specify network configuration and devices. eg. Static IP and info, DHCP, etc. |

The /etc/sysconfig/network  file

* The /etc/sysconfig/network file is a global (across all network cards) configuration file. It allows us to define whether we want networking (NETWORKING=yes|no), what the hostname should be (HOSTNAME=) and which gateway to use (GATEWAY=).
* Note that this file contains no settings at all in a default RHEL7 install (with networking enabled).

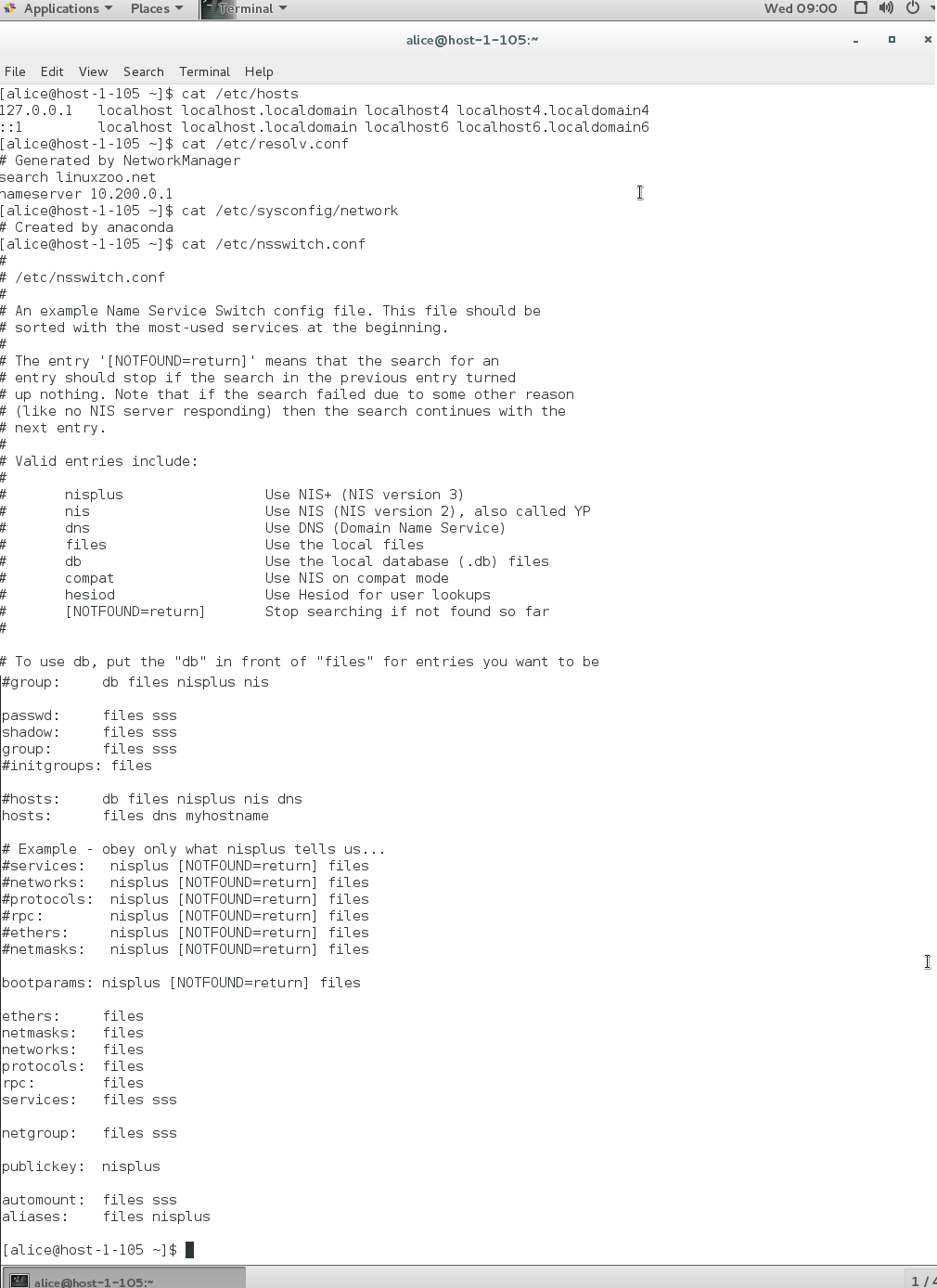
**The /etc/hosts  file**

The main purpose of /etc/hosts configuration file is to resolve hostnames that cannot be resolved any other way. It can also be used to resolve hostnames on small networks with no DNS server.

**The /etc/resolv.conf file**

This file is used for configuring the DNS (Domain Name System) resolver library. The resolv.conf configuration file contains information parameters used by the DNS resolver. The DNS resolver allows for the operating system to translate domain names into IP addresses.

Example:

 Typical default contents:

| **Directive** | **Description** |
| --- | --- |
| auto | Indicates the device should be setup at boot time |
| lo | Loopback interface |
| iface | Interface |
| eth0 | Ethernet device 0, typically the primary network adaptor |
| inet | Indicates network adaptor has an IPv4 address space |
| dhcp | Network adaptor gets its configuration from a DHCP server |
| static | Indicates the adaptor uses fixed IP configuration |
| address | The IP address of the host |
| netmask | Network subnet mask |
| gateway | Gateway Address |
| network | The network portion of the IP address |
| nameserver | The IP of a DNS |

**💡Tips:**

* The loopback (lo) interface will have an IP address of 127.0.0.1, which represents the host itself. Suppose you want to open a web page running on the same Linux server you are on. You could open http://127.0.0.1 in your web browser. That IP address won’t be accessible over the network.
* The ethernet 0 (eth0) interface is typically the connection to the local network. Even if you are running Linux in a virtual machine (VM), you’ll still have an eth0 interface that connects to the physical network interface of the host. Most commonly, you should ensure that eth0 is in an IP state and has an IP address so that you can communicate with the local network and likely over the Internet.