Ping Command Syntax

ping [**-t**] [**-a**] [**-n** *count*] [**-l** *size*] [**-f**] [**-i** *TTL*] [**-v** *TOS*] [**-r** *count*] [**-s** *count*] [**-w** *timeout*] [**-R**] [**-S** *srcaddr*] [**-4**] [**-6**] *target* [**/?**]

Tip: See How To Read Command Syntax if you're not sure how to interpret the ping command syntax above.

-t = Using this option will ping the *target* until you force it to stop using Ctrl-C.

-a = This ping command option will resolve, if possible, the hostname of an IP address *target*.

-n *count* = This option sets the number of ICMP Echo Request messages to send. If you execute the ping command without this option, four requests will be sent.

-l *size* = Use this option to set the size, in bytes, of the echo request packet from 32 to 65,527. The ping command will send a 32 byte echo request if you don't use the **-l** option.

-f = Use this ping command option to prevent ICMP Echo Requests from being fragmented by routers between you and the *target*. The **-f** option is most often used to troubleshoot Path Maximum Transmission Unit (PMTU) issues.

-i *TTL* = This option sets the Time to Live (TTL) value, the maximum of which is 255.

-v *TOS* = This option allows you to set a Type of Service (TOS) value. Beginning in Windows 7, this option no longer functions but still exists for compatibility reasons.

-r *count* = Use this ping command option to specify the number of hops between the your computer and the *target* computer or device that you'd like to be recorded and displayed. The maximum value for *count* is 9 so use the tracert command instead if you're interested in viewing all hops between two devices.

-s *count* = Use this option to report the time, in Internet Timestamp format, that each echo request is received and echo reply is sent. The maximum value for *count* is 4 meaning that only the first four hops can be time stamped.

-w *timeout* = Specifying a *timeout* value when executing the ping command adjusts the amount of time, in milliseconds, that ping waits for each reply. If you don't use the **-w** option, the default timeout value is used which is 4000, or 4 seconds.

-R = This option tells the ping command to trace the round trip path.

-S *srcaddr* = Use this option to specify the source address.

-4 = This forces the ping command to use IPv4 only but is only necessary if *target* is a hostname and not an IP address.

-6 = This forces the ping command to use IPv6 only but as with the **-4** option, is only necessary when pinging a hostname.

target = This is the destination you wish to ping, either an IP address or a hostname.

/? = Use the help switch with the ping command to show detailed help about the command's several options.

Note: The **-f**, **-v**, **-r**, **-s**, **-j**, and **-k** options work when pinging IPv4 addresses only. The **-R** and **-S** options only work with IPv6.

Other less commonly used switches for the ping command exist including [**-j** *host-list*] and [**-k** *host-list*]. Execute **ping /?** from the Command Prompt for more information on these two options.

Tip: Save all that ping command output to a file using a redirection operator. See How To Redirect Command Output to a File for instructions or see my Command Prompt Tricks list for more tips.

Ping Command Examples

```
ping -n 5 -l 1500 www.google.com
```

In this example, the ping command is used to ping the hostname *www.google.com*. The **-n** option tells the ping command to send *5* ICMP Echo Requests instead of the default of 4 and the **-l** option sets the packet size for each request to *1500* bytes instead of the default of 32 bytes. The result displayed in the Command Prompt window will look something like this:

```
Pinging www.google.com [74.125.224.82] with 1500 bytes of data:
Reply from 74.125.224.82: bytes=1500 time=68ms TTL=52
Reply from 74.125.224.82: bytes=1500 time=68ms TTL=52
Reply from 74.125.224.82: bytes=1500 time=65ms TTL=52
Reply from 74.125.224.82: bytes=1500 time=66ms TTL=52
Reply from 74.125.224.82: bytes=1500 time=70ms TTL=52

Ping statistics for 74.125.224.82:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 65ms, Maximum = 70ms, Average = 67ms
```

The *0% loss* reported under *Ping statistics for 74.125.224.82* tells me that each ICMP Echo Request message sent to *www.google.com* was returned. This means that, as far as my network connection goes, I can communicate with Google's website just fine.

```
ping 127.0.0.1
```

In the above example, I'm pinging *127.0.0.1*, also called the IPv4 localhost IP address or IPv4 loopback IP address, without options.

Using the ping command to ping *127.0.0.1* is an excellent way to test that Windows' network features are working properly but it says nothing about your own network hardware or your connection to any other computer or device. The IPv6 version of this test would be **ping ::1**.

```
ping -a 192.168.1.22
```

In this example I'm asking the ping command to find the hostname assigned to the *192.168.1.22* IP address but otherwise ping it as normal.

```
Pinging J3RTY22 [192.168.1.22] with 32 bytes of data:
Reply from 192.168.1.22: bytes=32 time<1ms TTL=64
Reply from 192.168.1.22: bytes=32 time<1ms TTL=64
Reply from 192.168.1.22: bytes=32 time=1ms TTL=64
Reply from 192.168.1.22: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.1.22:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
```

```
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

As you can see, the ping command resolved the IP address I entered, *192.168.1.22*, as the hostname *J3RTY22* and then executed the remainder of the ping with default settings.

```
ping -t -6 SERVER
```

In this example, I force the ping command to use IPv6 with the **-6** option and continue to ping *SERVER* indefinitely with the **-t** option.

```
Pinging SERVER [fe80::fd1a:3327:2937:7df3%10] with 32 bytes of data:
Reply from fe80::fd1a:3327:2937:7df3%10: time=1ms
Reply from fe80::fd1a:3327:2937:7df3%10: time<1ms
Reply from fe80::fd1a:3327:2937:7df3%10: time<1ms
Reply from fe80::fd1a:3327:2937:7df3%10: time<1ms
Reply from fe80::fd1a:3327:2937:7df3%10: time<1ms
Reply from fe80::fd1a:3327:2937:7df3%10: time<1ms
Reply from fe80::fd1a:3327:2937:7df3%10: time<1ms

Ping statistics for fe80::fd1a:3327:2937:7df3%10:
    Packets: Sent = 7, Received = 7, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
Control-C
^C
```

I interrupted the ping manually with Ctrl-C after seven replies. Also, as you can see, the **-6** option produced IPv6 addresses.

Tip: The number after the % in the replies generated in this ping command example is the IPv6 Zone ID, which most often indicates the network interface used. You can generate a table of Zone IDs matched with your network interface names by executing **netsh interface ipv6 show interface**. The IPv6 Zone ID is the number in the *Idx* column.

IPCONFIG

Configure IP (*internet protocol* configuration)

```
Syntax

  IPCONFIG /all          Display full configuration information.

  IPCONFIG /release [adapter]
                         Release the IP address for the specified adapter.

  IPCONFIG /renew [adapter]
                         Renew the IP address for the specified adapter.

  IPCONFIG /flushdns     Purge the DNS Resolver cache.

  IPCONFIG /registerdns Refresh all DHCP leases and re-register DNS names.

  IPCONFIG /displaydns  Display the contents of the DNS Resolver Cache.

  IPCONFIG /showclassid adapter
                         Display all the DHCP class IDs allowed for adapter.
```

```
IPCONFIG /setclassid  adapter [classid]
                      Modify the dhcp class id.
```

If the Adapter name contains spaces, use quotes: "Adapter Name"
wildcard characters * and ? allowed, see the examples below

The default is to display only the IP address, subnet mask and default gateway for each adapter bound to TCP/IP.

For Release and Renew, if no adapter name is specified, then the IP address leases for all adapters bound to TCP/IP will be released or renewed.

For Setclassid, if no ClassId is specified, then the ClassId is removed.

```
Examples:
    > ipconfig                    ... Show information.
    > ipconfig /all               ... Show detailed information
    > ipconfig /renew             ... renew all adapters
    > ipconfig /renew EL*         ... renew any connection that has its
                                      name starting with EL

    > ipconfig /release *Con*     ... release all matching connections,
                                      eg. "Local Area Connection 1" or
                                          "Local Area Connection 2"

    > ipconfig /setclassid "Local Area Connection" TEST
                                  ... set the DHCP class ID for the
                                      named adapter to = TEST
```

NSLOOKUP

Start the Command Interpreter

Type the following into the Run dialog box:

cmd

Click the OK button.

Start Nslookup

3. Type:

nslookup

Press Enter on the keyboard.

Check What Options Are Available With Help or ?

Type:

4. help

or, type:

?

Press Enter on the keyboard.

To Check Mail or MX Records

5. To Check DNS Mail or MX Records for a specific domain, it is easiest to first set the parameters, then to make the actual query, in two separate steps. (More advanced queries can have more than two parameters set across several commands.) To query for MX Records, use the SET TYPE option. So, to check the MX Records for

the domain, spiceworks.com, type:

set type=MX

Press ENTER on the keyboard. This tells nslookup that any subsequent queries are to return MX record information. This is not just a one time setting, as all queries for this nslookup session will remain for the type set, until the type is changed.

To finish the query for MX Records for the spiceworks.com domain, next type:

spiceworks.com

Press ENTER on the keyboard.

To Check All or ANY Records

To check all or ANY DNS Records for a specific domain, it is easiest to first set the parameters, then to make the actual query, in two separate steps. This is similar to the method mentioned above, in "To Check Mail or MX Records." To check for ANY Records for the domain, spiceworks.com, type:

6.

set type=ANY

Press ENTER on the keyboard. Then type:

spiceworks.com

Press ENTER on the keyboard.

To Change The Default DNS Server

Something which is very much misunderstood by most entities is that, at times, propagation can be bypassed. This is helpful to verify DNS changes without having to wait for propagation. One method for bypassing propagation is

7. to change the nslookup session's default DNS server to the authoritative DNS server of the domain which is to be queried. This does not always work as expected, for various reasons, but it is usually worthwhile to try.

In previous examples, the authoritative name server, or DNS server, for the domain spiceworks.com has been

ns1.onr.com. So, to change the default DNS server to this name server, type:

server ns1.onr.com

Press ENTER on the keyboard.

To Receive Verbose Feedback - Debug

To receive verbose or debugging information for each query, use the SET DEBUG option. To do this, type:

set debug

Press ENTER on the keyboard.

Now that the debug option has been set, subsequent queries will display more information about what queries and communication is actually taking place. To see an example of the debugging information, type:

8.

spiceworks.com

Press ENTER on the keyboard. You should see more output on the screen which gives more details regarding the query.

To see even more verbose output, use the SET D2 option. To do this, type:

set d2

Press ENTER on the keyboard. Subsequent queries will yield a lot more output.

Tracert Command Syntax

tracert [**-d**] [**-h** *MaxHops*] [**-w** *TimeOut*] [**-4**] [**-6**] *target* [**/?**]

Tip: See How To Read Command Syntax if you're having a hard time understanding the tracert syntax above.

-d = This option prevents tracert from resolving IP addresses to hostnames, often resulting in much faster results.

-h *MaxHops* = This tracert option specifies the maximum number of hops in the search for the *target*. If you do not specify *MaxHops*, and *target* has not been found by 30 hops, tracert will stop looking.

-w *TimeOut* = You can specify the time, in milliseconds, to allow each reply before timeout using this tracert option.

-4 = This option forces tracert to use IPv4 only.

-6 = This option forces tracert to use IPv6 only.

target = This is the destination, either an IP address or hostname.

/? = Use the help switch with the tracert command to show detailed help about the command's several options.

Other less commonly used options for the tracert command also exist including [**-j** *HostList*], [**-R**], and [**-S** *SourceAddress*]. Use the help switch with the tracert command for more information on these options.

Tip: Save the lengthy results of a tracert command to a file with a redirection operator. Take a look at How To Redirect Command Output to a File for help or see Command Prompt Tricks for this and other helpful tips.

Tracert Command Examples

```
tracert 192.168.1.1
```

In the above example, the tracert command is used to show the path from the networked computer on which the tracert command is being executed to a network device, in this case a router on a local network, that's assigned the *192.168.1.1* IP address. The result displayed on screen will look something like this:

```
Tracing route to 192.168.1.1 over a maximum of 30 hops

  1    <1 ms    <1 ms    <1 ms  192.168.1.254
  2    <1 ms    <1 ms    <1 ms  192.168.1.1

Trace complete.
```

In this example, you can see that tracert found a network device using the IP address of *192.168.1.254*, let's say a network switch, followed by the destination, *192.168.1.1*, the router.

```
tracert www.google.com
```

Using the tracert command as shown above, we're asking tracert to show us the path from the local computer all the way to the network device with the hostname *www.google.com*.

```
Tracing route to www.l.google.com [209.85.225.104]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  10.1.0.1
  2    35 ms    19 ms    29 ms  98.245.140.1
  3    11 ms    27 ms     9 ms  te-0-3.dnv.comcast.net [68.85.105.201]
  ...
 13    81 ms    76 ms    75 ms  209.85.241.37
 14    84 ms    91 ms    87 ms  209.85.248.102
 15    76 ms   112 ms    76 ms  iy-f104.1e100.net [209.85.225.104]
```

```
Trace complete.
```

In this example we can see that tracert identified fifteen network devices including our router at *10.1.0.1* and all the way through to the *target* of *www.google.com* which we now know uses the public IP address of *209.85.225.104*.

Note: I excluded hops 4 through 12 above just to keep the example simple. If you were executing a real tracert, those results would all show up on screen.

```
tracert -d www.yahoo.com
```

In this final tracert command example, you can see that I'm again requesting the path to a website, this time *www.yahoo.com*, but now I'm preventing tracert from resolving hostnames by using the *-d* option.

```
Tracing route to any-fp.wa1.b.yahoo.com [209.191.122.70]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  10.1.0.1
  2    29 ms    23 ms    20 ms  98.245.140.1
  3     9 ms    16 ms    14 ms  68.85.105.201
  ...
 13    98 ms    77 ms    79 ms  209.191.78.131
 14    80 ms    88 ms    89 ms  68.142.193.11
 15    77 ms    79 ms    78 ms  209.191.122.70

Trace complete.
```

In this example we can see that tracert again identified fifteen network devices including our router at *10.1.0.1* and all the way through to the *target* of *www.yahoo.com* which we can assume uses the public IP address of *209.191.122.70*.

As you can see, tracert did not resolve any hostnames this time which I assure you significantly sped up the process.

Tracert Command Availability

The tracert command is available from within the Command Prompt in all Windows operating systems including Windows 8, Windows 7, Windows Vista, Windows XP, and older versions of Windows as well.

Note: The availability of certain tracert command switches and other tracert command syntax may differ from operating system to operating system.

Tracert Related Commands

The tracert command is often used with other networking related Command Prompt commands like ping, ipconfig, netstat, nslookup, and others.

Netstat Command Syntax

netstat [**-a**] [**-b**] [**-e**] [**-f**] [**-n**] [**-o**] [**-p** *protocol*] [**-r**] [**-s**] [**-t**] [**-x**] [**-y**] [*time_interval*] [**/?**]

Tip: See How To Read Command Syntax if you're not sure how to read the netstat command syntax above.

Execute the netstat command alone to show a relatively simple list of all active TCP connections which, for each one, will show the local IP address (your computer), the foreign IP address (the other computer or network device), along with their respective port numbers, as well as the TCP state.

-a = This switch displays active TCP connections, TCP connections with the listening state, as well as UDP ports that are being listened to.

-b = This netstat switch is very similar to the **-o** switch listed below, but instead of displaying the PID, will display the process's actual file name. Using **-b** over **-o** might seem like it's saving you a step or two but using it can sometimes greatly extend the time it takes netstat to fully execute.

-e = Use this switch with the netstat command to show statistics about your network connection. This data includes bytes, unicast packets, non-unicast packets, discards, errors, and unknown protocols received and sent since the connection was established.

-f = The **-f** switch will force the netstat command to display the Fully Qualified Domain Name (FQDN) for each foreign IP addresses when possible.

-n = Use the **-n** switch to prevent netstat from attempting to determine host names for foreign IP addresses. Depending on your current network connections, using this switch could considerably reduce the time it takes for netstat to fully execute.

-o = A handy option for many troubleshooting tasks, the **-o** switch displays the process identifier (PID) associated with each displayed connection. See the example below for more about using **netstat -o**.

-p = Use the **-p** switch to show connections or statistics only for a particular *protocol*. You can not define more than one *protocol* at once, nor can you execute netstat with **-p** without defining a *protocol*.

protocol = When specifying a *protocol* with the **-p** option, you can use **tcp**, **udp**, **tcpv6**, or **udpv6**. If you use **-s** with **-p** to view statistics by protocol, you can use **icmp**, **ip**, **icmpv6**, or **ipv6** in addition to the first four I mentioned.

-r = Execute netstat with **-r** to show the IP routing table. This is the same as using the route command to execute **route print**.

-s = The **-s** option can be used with the netstat command to show detailed statistics by protocol. You can limit the statistics shown to a particular protocol by using the **-s** option and specifying that *protocol*, but be sure to use **-s** before **-p** *protocol* when using the switches together.

-t = Use the **-t** switch to show the current TCP chimney offload state in place of the typically displayed TCP state.

-x = Use the **-x** option to show all NetworkDirect listeners, connections, and shared endpoints.

-y = The **-y** switch can be used to show the TCP connection template for all connection. You cannot use **-y** with any other netstat option.

time_interval = This is the time, in seconds, that you'd like the netstat command to re-execute automatically, stopping only when you use Ctrl-C to end the loop.

/? = Use the help switch to show details about the netstat command's several options.

Tip: Make all that netstat information in the command line easier to work with by outputting what you see on the screen to a text file using a redirection operator. See How To Redirect Command Output to a File for complete instructions.

Netstat Command Examples

netstat -f

In this first example, I execute netstat to show all active TCP connections. However, I do want to see the computers I'm connected to in FQDN format [**-f**] instead of a simple IP address.

Here's an example of what you might see:

```
Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:5357         VM-Windows-7:49229     TIME_WAIT
  TCP    127.0.0.1:49225        VM-Windows-7:12080     TIME_WAIT
  TCP    192.168.1.14:49194     75.125.212.75:http     CLOSE_WAIT
  TCP    192.168.1.14:49196     a795sm.avast.com:http  CLOSE_WAIT
  TCP    192.168.1.14:49197     a795sm.avast.com:http  CLOSE_WAIT
  TCP    192.168.1.14:49230     TIM-PC:wsd             TIME_WAIT
  TCP    192.168.1.14:49231     TIM-PC:icslap          ESTABLISHED
  TCP    192.168.1.14:49232     TIM-PC:netbios-ssn     TIME_WAIT
  TCP    192.168.1.14:49233     TIM-PC:netbios-ssn     TIME_WAIT
  TCP    [::1]:2869             VM-Windows-7:49226     ESTABLISHED
  TCP    [::1]:49226            VM-Windows-7:icslap    ESTABLISHED
```

As you can see, I had 11 active TCP connections at the time I executed netstat. The only protocol (in the *Proto* column) listed is TCP, which was expected because I did not use **-a**.

You can also see three sets of IP addresses in the *Local Address* column - my actual IP address of *192.168.1.14* and both IPv4 and IPv6 versions of my loopback addresses, along with the port each connection is using. The *Foreign Address* column lists the FQDN (*75.125.212.75* didn't resolve for some reason) along with that port as well.

Finally, the *State* column lists the TCP state of that particular connection.

netstat -o

In this example, I want to run netstat normally so it only shows active TCP connections, but I also want to see the corresponding process identifier [**-o**] for each connection so I can determine which program on my computer initiated each one.

Here's what my computer displayed:

```
Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    192.168.1.14:49194     75.125.212.75:http     CLOSE_WAIT      2948
  TCP    192.168.1.14:49196     a795sm:http            CLOSE_WAIT      2948
  TCP    192.168.1.14:49197     a795sm:http            CLOSE_WAIT      2948
```

You probably noticed the new *PID* column. In this case, the PIDs are all the same, meaning that the same program on my computer opened these connections.

To determine what program is represented by the PID of *2948* on my computer, all I have to do is open Task Manager, click on the *Processes* tab, and note the *Image Name* listed next to the PID I'm looking

for in the *PID* column.[1]

Using the netstat command with the **-o** option can be very helpful when tracking down which program is using too big a share of your bandwidth. It can also help locate the destination where some kind of malware, or even an otherwise legitimate piece of software, might be sending information without your permission.

Note: While this and the previous example were both run on the same computer, and within just a minute of each other, you can see that the list of active TCP connections is considerably different. This is because your computer is constantly connecting to, and disconnecting from, various other devices on your network and over the Internet.

```
netstat -s -p tcp -f
```

In this third example, I want to see protocol specific statistics [**-s**] but not all of them, just TCP stats [**-p** *tcp*]. I also want the foreign addresses displayed in FQDN format [**-f**].

This is what the netstat command, as shown above, produced on my computer:

```
TCP Statistics for IPv4

  Active Opens                            = 77
  Passive Opens                           = 21
  Failed Connection Attempts              = 2
  Reset Connections                       = 25
  Current Connections                     = 5
  Segments Received                       = 7313
  Segments Sent                           = 4824
  Segments Retransmitted                  = 5

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:2869         VM-Windows-7:49235     TIME_WAIT
  TCP    127.0.0.1:2869         VM-Windows-7:49238     ESTABLISHED
  TCP    127.0.0.1:49238        VM-Windows-7:icslap    ESTABLISHED
  TCP    192.168.1.14:49194     75.125.212.75:http     CLOSE_WAIT
  TCP    192.168.1.14:49196     a795sm.avast.com:http  CLOSE_WAIT
  TCP    192.168.1.14:49197     a795sm.avast.com:http  CLOSE_WAIT
```

As you can see, various statistics for the TCP protocol are displayed, as are all active TCP connections at the time.

```
netstat -e -t 5
```

In this final example, I executed the netstat command to show some basic network interface statistics [**-e**] and I wanted these statistics to continually update in the command window every five seconds [**-t** *5*].

Here's what's produced on screen:

```
Interface Statistics

                           Received            Sent

Bytes                      22132338         1846834
Unicast packets               19113            9869
Non-unicast packets               0               0
Discards                          0               0
Errors                            0               0
```

```
Unknown protocols                  0
Interface Statistics

                           Received            Sent

Bytes                      22134630         1846834
Unicast packets               19128            9869
Non-unicast packets               0               0
Discards                          0               0
Errors                            0               0
Unknown protocols                 0
^C
```

Various pieces of information, which you can see here and that I listed in the **-e** syntax above, are displayed.

I only let the netstat command automatically execute one extra time, as you can see by the two tables in the result. Note the *^C* at the bottom, indicating that I used the Ctrl-C abort command to stop the rerunning of the command.

Netstat Command Availability

The netstat command is available from within the Command Prompt in most versions of Windows including Windows 8, Windows 7, Windows Vista, Windows XP, Windows Server operating systems, and some older versions of Windows too.

Note: The availability of certain netstat command switches and other netstat command syntax may differ from operating system to operating system.

Netstat Related Commands

The netstat command is often used with other networking related Command Prompt commands like nslookup, ping, tracert, ipconfig, and others.