Using The traceroute Command

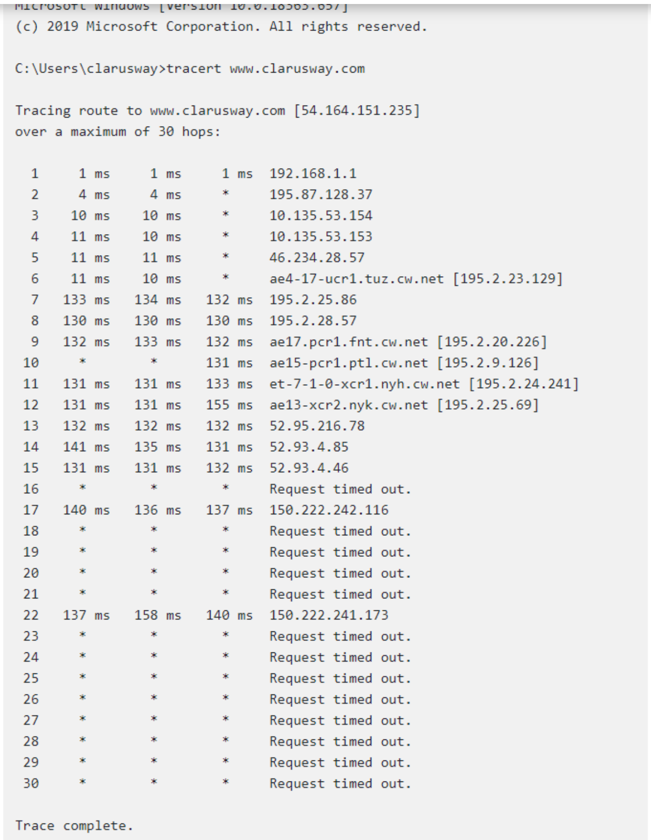
Using traceroute

Where do all those packets really go when we send them over the Internet? And, how do all the packets actually get to their destinations? Well, we can use the **TCP/IP traceroute** (*tracert with Windows*) command-line utility to help us answer both questions because its output will show us every router interface a TCP/IP packet passes through on the way to its destination.

Traceroute (trace for short) displays the path a packet takes to get to a remote device by using something we call **IP packet Time to Live (TTL) time-outs** and **Internet Control Message Protocol (ICMP)** error messages. And it's also a handy tool for troubleshooting an internetwork because we can use it to figure out which router along a path through that internetwork happens to be causing a network failure when a certain destination machine or network is, or suddenly becomes, unreachable.

To use **tracert**, at a Windows command prompt, type `tracert`, a space, and the `Domain Name Service (DNS) name` or `IP address` of the host machine to which you want to find the route. The **tracert utility** will respond with a list of all the DNS names and IP addresses of the routers that the packet is passing through on its way. Plus, tracert uses TTL to indicate the time it takes for each attempt.

Following is the *tracert output* from a local pc to clarusway.com server:

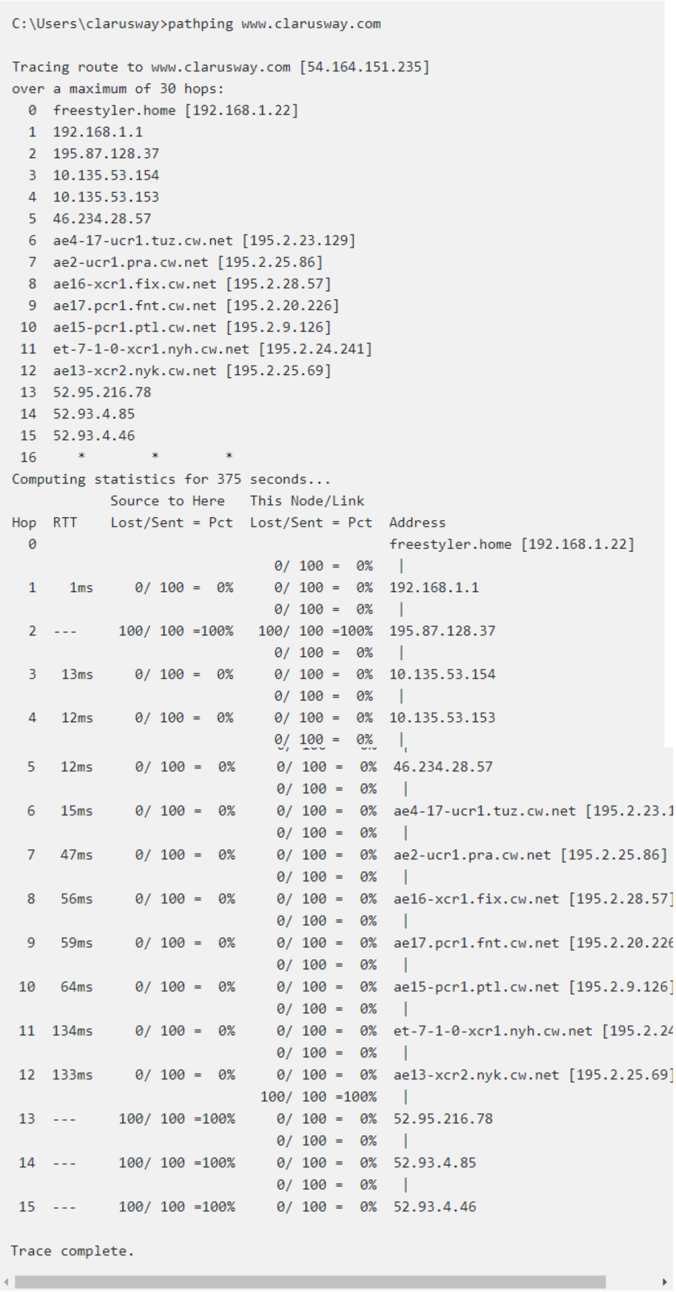


```
Microsoft Windows [Version 10.0.18363.657]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\clarusway>tracert www.clarusway.com

Tracing route to www.clarusway.com [54.164.151.235]
over a maximum of 30 hops:

  1     1 ms     1 ms     1 ms  192.168.1.1
  2     4 ms     4 ms     *     195.87.128.37
  3    10 ms    10 ms     *     10.135.53.154
  4    11 ms    10 ms     *     10.135.53.153
  5    11 ms    11 ms     *     46.234.28.57
  6    11 ms    10 ms     *     ae4-17-ucr1.tuz.cw.net [195.2.23.129]
  7   133 ms   134 ms   132 ms  195.2.25.86
  8   130 ms   130 ms   130 ms  195.2.28.57
  9   132 ms   133 ms   132 ms  ae17.pcr1.fnt.cw.net [195.2.20.226]
 10     *        *      131 ms  ae15-pcr1.ptl.cw.net [195.2.9.126]
 11   131 ms   131 ms   133 ms  et-7-1-0-xcr1.nyh.cw.net [195.2.24.241]
 12   131 ms   131 ms   155 ms  ae13-xcr2.nyk.cw.net [195.2.25.69]
 13   132 ms   132 ms   132 ms  52.95.216.78
 14   141 ms   135 ms   131 ms  52.93.4.85
 15   131 ms   131 ms   132 ms  52.93.4.46
 16     *        *        *     Request timed out.
 17   140 ms   136 ms   137 ms  150.222.242.116
 18     *        *        *     Request timed out.
 19     *        *        *     Request timed out.
 20     *        *        *     Request timed out.
 21     *        *        *     Request timed out.
 22   137 ms   158 ms   140 ms  150.222.241.173
 23     *        *        *     Request timed out.
 24     *        *        *     Request timed out.
 25     *        *        *     Request timed out.
 26     *        *        *     Request timed out.
 27     *        *        *     Request timed out.
 28     *        *        *     Request timed out.
 29     *        *        *     Request timed out.
 30     *        *        *     Request timed out.

Trace complete.
```

You see that the packet bounces through several routers before arriving at its destination. This utility is useful if you are having problems reaching a web server on the Internet and you want to know if a wide area network (WAN) link is down or if the server just isn't responding. What this means to you is that, basically, wherever the trace stops is a great place to start troubleshooting. Notice in the output the *ms*. This is the latency of each hop, meaning the delay. `Tracert` (or `traceroute`) is a great troubleshooting tool you can use to find out where your network bottlenecks are.

If you use `traceroute` or `tracert` and receive an asterisk, this indicates that the attempt to reach that router took longer than the default time-out value. This is very good to know because it can mean that either the router is extremely busy or a particular link is slow. Another reason for getting an asterisk could be that the administrator has disabled ICMP on the router that the packet is trying to hop through because of security reasons. It happens to be a typical strategic move done on the routers that interface to the ISP to conceal their actual location so bad guys can't hack into them and therefore into your internetwork.

In addition to `traceroute` and `tracert`, you can use `pathping` (for Windows), which is a lot like `traceroute`:



```
C:\Users\clarusway>pathping www.clarusway.com

Tracing route to www.clarusway.com [54.164.151.235]
over a maximum of 30 hops:
  0  freestyler.home [192.168.1.22]
  1  192.168.1.1
  2  195.87.128.37
  3  10.135.53.154
  4  10.135.53.153
  5  46.234.28.57
  6  ae4-17-ucr1.tuz.cw.net [195.2.23.129]
  7  ae2-ucr1.pra.cw.net [195.2.25.86]
  8  ae16-xcr1.fix.cw.net [195.2.28.57]
  9  ae17.pcr1.fnt.cw.net [195.2.20.226]
 10  ae15-pcr1.ptl.cw.net [195.2.9.126]
 11  et-7-1-0-xcr1.nyh.cw.net [195.2.24.241]
 12  ae13-xcr2.nyk.cw.net [195.2.25.69]
 13  52.95.216.78
 14  52.93.4.85
 15  52.93.4.46
 16     *        *        *
Computing statistics for 375 seconds...
            Source to Here   This Node/Link
Hop  RTT    Lost/Sent = Pct  Lost/Sent = Pct  Address
  0                                           freestyler.home [192.168.1.22]
                                0/ 100 =  0%   |
  1    1ms     0/ 100 =  0%     0/ 100 =  0%  192.168.1.1
                                0/ 100 =  0%   |
  2  ---     100/ 100 =100%   100/ 100 =100%  195.87.128.37
                                0/ 100 =  0%   |
  3   13ms     0/ 100 =  0%     0/ 100 =  0%  10.135.53.154
                                0/ 100 =  0%   |
  4   12ms     0/ 100 =  0%     0/ 100 =  0%  10.135.53.153
                                0/ 100 =  0%   |
  5   12ms     0/ 100 =  0%     0/ 100 =  0%  46.234.28.57
                                0/ 100 =  0%   |
  6   15ms     0/ 100 =  0%     0/ 100 =  0%  ae4-17-ucr1.tuz.cw.net [195.2.23.1
                                0/ 100 =  0%   |
  7   47ms     0/ 100 =  0%     0/ 100 =  0%  ae2-ucr1.pra.cw.net [195.2.25.86]
                                0/ 100 =  0%   |
  8   56ms     0/ 100 =  0%     0/ 100 =  0%  ae16-xcr1.fix.cw.net [195.2.28.57]
                                0/ 100 =  0%   |
  9   59ms     0/ 100 =  0%     0/ 100 =  0%  ae17.pcr1.fnt.cw.net [195.2.20.226
                                0/ 100 =  0%   |
 10   64ms     0/ 100 =  0%     0/ 100 =  0%  ae15-pcr1.ptl.cw.net [195.2.9.126]
                                0/ 100 =  0%   |
 11  134ms     0/ 100 =  0%     0/ 100 =  0%  et-7-1-0-xcr1.nyh.cw.net [195.2.24
                                0/ 100 =  0%   |
 12  133ms     0/ 100 =  0%     0/ 100 =  0%  ae13-xcr2.nyk.cw.net [195.2.25.69]
                              100/ 100 =100%   |
 13  ---     100/ 100 =100%     0/ 100 =  0%  52.95.216.78
                                0/ 100 =  0%   |
 14  ---     100/ 100 =100%     0/ 100 =  0%  52.93.4.85
                                0/ 100 =  0%   |
 15  ---     100/ 100 =100%     0/ 100 =  0%  52.93.4.46

Trace complete.
```

Using the ipconfig Utility

With the new Mac, Windows 10, and Windows Server 2016 operating systems, you can now see the IPv6 configuration because IPv6 is enabled by default. The output of the `ipconfig` command provides the basic routed protocol information on your machine. From a DOS prompt, type `ipconfig`, and you'll see something like this:

```
C:\Users\clarusway>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:
   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 3:
   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:
   Connection-specific DNS Suffix  . : home
   Link-local IPv6 Address . . . . . : fe80::19ac:8efb:2c6e:f512%10
   IPv4 Address. . . . . . . . . . . : 192.168.1.22
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter Teredo Tunneling Pseudo-Interface:
   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2001:0:2851:782c:148e:f3fd:6aff:55b8
   Link-local IPv6 Address . . . . . : fe80::148e:f3fd:6aff:55b8%17
   Default Gateway . . . . . . . . . : ::
```

You can see that Ethernet adapter shows up first, and it has an IP address, a mask, and a default gateway plus an IPv6 address and a DNS suffix. The next configured interface is the wireless local area network (LAN) adapter, which has an IP address, a mask, a default gateway, an IPv6 address, and the IPv6 default gateway as well.

The next adapters are disconnected because they are logical interfaces and are not being used. But just in case the `ipconfig` command doesn't provide enough information for you, try the `ipconfig /all` command. Here's the beginning of that output:

```
C:\Users\clarusway>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : clarusway
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : home

Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Intel(R) I211 Gigabit Network Connection
   Physical Address. . . . . . . . . : 9C-5C-8E-CE-D9-C9
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter Ethernet 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Intel(R) Ethernet Connection (2) I219-V
   Physical Address. . . . . . . . . : 9C-5C-8E-CE-D9-CA
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
```

As you can see, it's more of the same—a whole lot more. The most important thing that you can see the hardware information about each interface, including the Media Access Control (MAC) address. Also significant is that you can see the Dynamic Host Configuration Protocol (DHCP) lease times and DNS addresses now.

There are two more valuable options you need to use with the `ipconfig` command. They are `/release` and `/renew`.

When you change networks, you need to get the IP address of that subnet and/or virtual LAN (VLAN). Windows 10 works most of the time without doing anything, but sometimes you have to renew the IP configuration when changing networks. For that, just type `ipconfig /renew` from a command prompt, and if you're connected to a DHCP server that's available. Now, if it still doesn't work, you'll need to release and renew your TCP/IP settings. To release your current DHCP TCP/IP information, you must elevate your command prompt or you'll get this warning:

```
C:\Users\clarusway>ipconfig /release
The requested operation requires elevation.
```

In order to avoid this, choose Start > All Programs > Accessories > Command Prompt, right-click, and choose Run As Administrator.

Once your command prompt has been duly elevated, you can use the `ipconfig /release` command and then the `ipconfig /renew` command to get new TCP/IP information for your host.

Using the ifconfig Utility

There is a utility in Linux/Unix/Mac that will give you information similar to what `ipconfig` shows. It's called `ifconfig` (short for interface configuration). Although `ipconfig` and `ifconfig` show similar information, there are major differences between these two utilities.

The `ipconfig` utility is mainly used to view the TCP/IP configuration for a computer. You can use `ifconfig` to do the same thing, but `ifconfig` can also be used to configure a protocol or a particular network interface.

The general syntax of the `ifconfig` command is as follows:

```
ifconfig interface [address [parameters]]
```

The interface parameter equals the Unix name of the interface, such as eth0. If the optional address parameter is specified, the `ifconfig` command sets the IP address for the interface to the address you've specified. When the `ifconfig` command is used by itself with no parameters, all configured interfaces will be reported on. But if only the interface name is specified, you'll get output that looks like this:

```
# ifconfig eth0
eth0 Link encap 10Mbps Ethernet HWaddr 00:00:C0:90:B3:42
inetaddr 172.16.0.2 Bcast 172.16.0.255 Mask 255.255.255.0 UP
BROADCAST RUNNING MTU 1500 Metric 0
    RX packets 3136 errors 217 dropped 7 overrun 26
    TX packets 1752 errors 25 dropped 0 overrun 0
```

Looking at this, we can see that the eth0 interface is a 10 Mbps Ethernet interface. The interface's MAC and IP address information is displayed in this output as well. And, although not shown in the output, the `ifconfig` tool can show you the DNS information configured on the host.

Using the iptables Utility

The **iptables firewall utility** is built for the Linux operating system. It is a command-line utility that uses what are called chains to allow or disallow traffic. When traffic arrives, `iptables` looks for a rule that addresses that traffic type, and if none exists, it will enforce the default rule. There are three different chain types:

1. **Input:** Controls behavior for incoming connections
2. **Forward:** Used for incoming connections that aren't being delivered locally
3. **Output:** Used for outgoing connections

You can set the default action to accept, drop, or reject, with the difference between reject and drop being that reject sends an error message back to the source.

Examples of `iptables`

- To block a connection from the device at `192.168.10.1`, use this command:

```
iptables -A INPUT -s 192.168.10.1 -j DROP
```

- To block all connections from all devices in the `172.16.0.0/16` network, use this command:

```
iptables -A INPUT -s 172.16.0.0/16 -j DROP
```

- Here is the command to block SSH connections from `10.110.61.5`:

```
iptables -A INPUT -p tcp --dport ssh -s 10.110.61.5 -j DROP
```

- Use this command to block SSH connections from any IP address:

```
iptables -A INPUT -p tcp --dport ssh -j DROP
```

- The following command is used to save the changes in Ubuntu:

```
sudo /sbin/iptables-save
```

- In Red Hat/CentOS, use either of the following commands:

```
/sbin/service iptables save
/etc/init.d/iptables save
```