Lab - Network Troubleshooting Tools

Objectives

Part 1: Launch the DEVASC VM

Part 2: Explore the ifconfig Troubleshooting Tool

Part 3: Explore the ping Troubleshooting Tool

Part 4: Explore the traceroute Troubleshooting Tool

Part 5: Explore the nslookup Troubleshooting Tool

Background / Scenario

In the effort to fix network connection issues, it is important for a developer to understand how to use basic network troubleshooting tools. These tools are used to determine what the connection problem might be.

Required Resources

- 1 PC with operating system of your choice
- Virtual Box or VMWare
- DEVASC Virtual Machine

Instructions

Part 1: Launch the DEVASC VM

If you have not already completed the **Lab - Install the Virtual Machine Lab Environment**, do so now. If you have already completed that lab, launch the **DEVASC VM**.

Part 2: Explore the ifconfig Troubleshooting Tool

The **ifconfig** tool is an application for use with UNIX-based operating systems such as Linux. A similar utility is available in Windows called ipconfig. These applications are used to manage network interfaces from the command line. You can use ifconfig to accomplish the following:

- Configure the IP address and subnet mask for an interface.
- Retrieve the status of network interfaces.
- Enable or disable network interfaces.
- Change the MAC address of a network interface.

Step 1: View the ifconfig options.

The **ifconfig** tool has many different options that can be added to the command to perform specific tasks.

a. Open a terminal window either directly from the desktop, or within VS Code.

b. Type **ifconfig --help** to see all of the available options for the command.

```
devasc@labvm:~$ ifconfig --help
Usage:
  ifconfig [-a] [-v] [-s] <interface> [[<AF>] <address>]
```

```
[add <address>[/<prefixlen>]]
[del <address>[/<prefixlen>]]
[[-]broadcast [<address>]]  [[-]pointopoint [<address>]]
[netmask <address>]  [dstaddr <address>]  [tunnel <address>]
[outfill <NN>] [keepalive <NN>]
[hw <HW> <address>]  [mtu <NN>]
[[-]trailers]  [[-]arp]  [[-]allmulti]
[multicast]  [[-]promisc]
[mem_start <NN>]  [io_addr <NN>]  [irq <NN>]  [media <type>]
[txqueuelen <NN>]
[[-]dynamic]
[up|down] ...
```

This is an overview of some of the more widely used options;

- **add** or **del** - This option allows you to add or delete IP addresses and their subnet mask (prefix length).
- **hw ether** - This is used to change the physical MAC address. This might be useful to, for example, change it to an easily recognizable name so that it stands out in logs for troubleshooting.
- **up** and **down** - These options are used to enable and disable interfaces. Be sure of which interface you are disabling. If it is the one you are using to remotely connect to a device, you will be disconnected!

Step 2: See the status of all interfaces.

a. Display the status of all the network interfaces in use by issuing the **ip addr** command by itself.

```
devasc@labvm:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen
1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
default qlen 1000
    link/ether 08:00:27:e9:3d:e6 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 85901sec preferred_lft 85901sec
    inet6 fe80::a00:27ff:fee9:3de6/64 scope link
       valid_lft forever preferred_lft forever
3: dummy0: <BROADCAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/ether e2:2b:24:96:98:b8 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.1/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.2/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.3/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.4/32 scope global dummy0
```

```
       valid_lft forever preferred_lft forever
    inet 192.0.2.5/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet6 fe80::e02b:24ff:fe96:98b8/64 scope link
       valid_lft forever preferred_lft forever
devasc@labvm:~$
```

From this output, we can tell a lot about the VM interfaces:

- There are 3 interfaces, the loopback interface (**lo**), **enp0s3**, and **dummy0**.
- **ether** shows the MAC address and that Ethernet is the link encapsulation.
- **inet** is the IP address, the subnet mask is shown in slash notation, and **brd** is the broadcast address.
- **UP** indicates that the interface is enabled.
- **mtu** is the Maximum Transmission Unit specifying the maximum number of bytes that the frame can be transmitted on this medium before being fragmented.

Part 3: Explore the ping Troubleshooting Tool

The **ping** tool is an application that is used to test network connectivity between devices. ping uses Internet Control Message Protocol (ICMP) to send packets to a device on the network and waits for the device to reply. ping reports network errors, packet loss, and the time to live (TTL), among other statistics.

Step 1: View the ping options.

Ping is only available in a terminal window or a command prompt.

a. Type **ping -help** to see all of the available options for the command.

```
devasc@labvm:~$ ping -help

Usage
  ping [options] <destination>

Options:
  <destination>      dns name or ip address
  -a                 use audible ping
  -A                 use adaptive ping
  -B                 sticky source address
  -c <count>         stop after <count> replies
  -D                 print timestamps
  -d                 use SO_DEBUG socket option
  -f                 flood ping
  -h                 print help and exit
  -I <interface>     either interface name or address
  -i <interval>      seconds between sending each packet
  -L                 suppress loopback of multicast packets
  -l <preload>       send <preload> number of packages while waiting replies
  -m <mark>          tag the packets going out
  -M <pmtud opt>     define mtu discovery, can be one of <do|dont|want>
  -n                 no dns name resolution
  -O                 report outstanding replies
  -p <pattern>       contents of padding byte
```

```
  -q                 quiet output
  -Q <tclass>        use quality of service <tclass> bits
  -s <size>          use <size> as number of data bytes to be sent
  -S <size>          use <size> as SO_SNDBUF socket option value
  -t <ttl>           define time to live
  -U                 print user-to-user latency
  -v                 verbose output
  -V                 print version and exit
  -w <deadline>      reply wait <deadline> in seconds
  -W <timeout>       time to wait for response

IPv4 options:
  -4                 use IPv4
  -b                 allow pinging broadcast
  -R                 record route
  -T <timestamp>     define timestamp, can be one of <tsonly|tsandaddr|tsprespec>

IPv6 options:
  -6                 use IPv6
  -F <flowlabel>     define flow label, default is random
  -N <nodeinfo opt>  use icmp6 node info query, try <help> as argument

For more details see ping(8).
devasc@labvm:~$
```

Step 2: Ping a host.

The **ping** tool has many different options that can be selected to customize how the communication should take place. Some of the options you can specify include:

- Specify how many ICMP echo requests you want to send.
- Identify the source IP address if there are multiple interfaces on the device.
- Indicate the amount of time to wait for a reply.
- Packet size, if you want to send larger packet sizes than the default 64 bytes. This can help to determine what is the maximum transmission unit (MTU) is.

a. ping www.cisco.com to see if it is reachable.

```
devasc@labvm:~$ ping -c 5 www.cisco.com
PING e2867.dsca.akamaiedge.net (23.66.161.25) 56(84) bytes of data.
64 bytes from a23-66-161-25.deploy.static.akamaitechnologies.com (23.66.161.25):
icmp_seq=1 ttl=49 time=58.4 ms
64 bytes from a23-66-161-25.deploy.static.akamaitechnologies.com (23.66.161.25):
icmp_seq=2 ttl=49 time=63.1 ms
64 bytes from a23-66-161-25.deploy.static.akamaitechnologies.com (23.66.161.25):
icmp_seq=3 ttl=49 time=61.2 ms
64 bytes from a23-66-161-25.deploy.static.akamaitechnologies.com (23.66.161.25):
icmp_seq=4 ttl=49 time=57.7 ms
64 bytes from a23-66-161-25.deploy.static.akamaitechnologies.com (23.66.161.25):
icmp_seq=5 ttl=49 time=57.6 ms

--- e2867.dsca.akamaiedge.net ping statistics ---
```

```
5 packets transmitted, 5 received, 0% packet loss, time 8153ms
rtt min/avg/max/mdev = 57.597/59.605/63.145/2.205 ms
devasc@labvm:~$
```

This ping specified a count of 5 packets.

The **ping** tool automatically does DNS resolution, returning 23.66.161.25 (your returned IP address may be different). Time to Live (TTL) for the received echo replies and round-trip times are also displayed. The final statistics confirm that 5 ICMP echo-request packets have been transmitted and 5 ICMP echo-reply packets have been received, achieving a 0% packet loss. Statistics about the minimum, average, maximum and standard deviation of the time it took for the packets to get to the destination and back are also displayed.

If you do not receive any replies from the destination doesn't necessarily mean that the host is offline or not reachable. It could mean that ICMP packets are being blocked by a firewall. It is a best practice to expose only the services needed to be available on the hosts in the network.

For IPv6 there exists a similar utility that is called **ping6** and is also available on most operating systems.

Part 4: Explore the traceroute Troubleshooting Tool

The **traceroute** tool displays the route that the packets take on their way to a destination. The Microsoft Windows alternative is called **tracert**. Observing the path network traffic takes from source to the destination is important for troubleshooting because routing loops and non-optimal paths can be detected and corrected.

traceroute uses ICMP packets to determine the path to the destination. The Time to Live (TTL) field in the IP packet header is used to avoid infinite loops on the network. For each hop or router that an IP packet goes through, the TTL field is decremented by one. When the TTL field value reaches 0, the packet is discarded avoiding infinite loops. Usually, the TTL field is set to its maximum value, 255, at the source of the traffic, because the host is trying to maximize the chances of that packet getting to its destination. traceroute reverses this logic, and gradually increments the TTL value, from 1 and keeps adding 1 to the TTL field on the next packet and so on. Setting a TTL value of 1 for the first packet, means the packet will be discarded on the first router. By default, most routers, send back to the source of the traffic an ICMP Time Exceeded packet informing it that the packet has reached a TTL value of 0 and had to be discarded. traceroute uses the information received from the router to figure out its IP address and hostname and also round-trip times.

For IPv6 there is an alternative called traceroute6 for UNIX-based operating systems and tracert6 for Microsoft Windows-based ones.

Step 1: View the traceroute options.

a. Type **traceroute --help** to see all of the available options for the command.

```
devasc@labvm:~$ traceroute --help
Usage: traceroute [OPTION...] HOST
Print the route packets trace to network host.

  -f, --first-hop=NUM        set initial hop distance, i.e., time-to-live
  -g, --gateways=GATES       list of gateways for loose source routing
  -I, --icmp                 use ICMP ECHO as probe
  -m, --max-hop=NUM          set maximal hop count (default: 64)
  -M, --type=METHOD          use METHOD (`icmp' or `udp') for traceroute
                             operations, defaulting to `udp'
  -p, --port=PORT            use destination PORT port (default: 33434)
  -q, --tries=NUM            send NUM probe packets per hop (default: 3)
      --resolve-hostnames    resolve hostnames
  -t, --tos=NUM              set type of service (TOS) to NUM
  -w, --wait=NUM             wait NUM seconds for response (default: 3)
```

```
  -?, --help                 give this help list
      --usage                give a short usage message
  -V, --version              print program version

Mandatory or optional arguments to long options are also mandatory or optional
for any corresponding short options.

Report bugs to <bug-inetutils@gnu.org>.
devasc@labvm:~$
```

Several options are available with **traceroute** including:

- Specify the TTL value of the first packet sent, 1 by default.
- Specify the maximum TTL value. By default, it will increase the TTL value up to 64 or until the destination is reached.
- Specify the source address in case there are multiple interfaces on the device.
- Specify Quality of Service (QoS) value in the IP header.
- Specify the packet length.

Step 2: Use traceroute to find the path to a web server.

Because of the way Virtual Box implements a NAT network, you cannot trace outside of your VM. You would need to change your VM to Bridged. But then, you would not be able to communicate with the CSR1000v in other labs. Therefore, we recommend leaving your VM in NAT mode.

However, you should be able to use the **traceroute** command on your local host. For Mac and Linux hosts, use the **traceroute** command. For Windows hosts, use the **tracert** command, as shown below. Open a command prompt on your local host and trace the route to www.netacad.com to see how many hops and how much time it takes to reach it. Your output will be different.

```
C:\> tracert www.netacad.com

Tracing route to e7792.dsca.akamaiedge.net [2600:1406:22:183::1e70]
over a maximum of 30 hops:

  1    43 ms    38 ms    36 ms  hsrp-2001-420-c0c8-1.cisco.com [2001:420:c0c8::1]
  2    48 ms    54 ms    40 ms  sjc05-sbb-gw1-twe1-0-13.cisco.com [2001:420:280:1aa::]
  3    39 ms    37 ms    38 ms  sjc05-rbb-gw1-por20.cisco.com [2001:420:41:116::]
  4    37 ms    38 ms    38 ms  sjc12-corp-gw1-ten1-3-0.cisco.com [2001:420:41:11c::1]
  5    39 ms    39 ms    45 ms  sjc12-dmzbb-gw1-vla777.cisco.com [2001:420:82:2::d]
  6    51 ms    39 ms    37 ms  sjc5-cbb-gw1-be92.cisco.com [2001:420:82:4e::]
  7    39 ms    39 ms    38 ms  sjc12-isp-gw2-ten0-0-0.cisco.com [2001:420:82:f::]
  8    78 ms    57 ms    65 ms  2001:1890:c00:6c01::eee7:a12
  9    44 ms    42 ms    47 ms  sj2ca81crs.ipv6.att.net
[2001:1890:ff:ffff:12:122:110:62]
 10    46 ms    46 ms    47 ms  2001:1890:ff:ffff:12:122:149:225
 11    43 ms    41 ms    43 ms  scaca401cts.ipv6.att.net
[2001:1890:ff:ffff:12:122:137:245]
 12    43 ms    43 ms    44 ms  2001:1890:fff:2180:12:120:13:178
 13    53 ms    54 ms    45 ms  2001:1890:1ff:2a80:12:120:183:64
 14    52 ms    42 ms    42 ms  g2600-1406-0022-0183-0000-0000-0000-
1e70.deploy.static.akamaitechnologies.com [2600:1406:22:183::1e70]
```

```
Trace complete.

devasc@labvm:~$
```

The output shows that there are 14 hops along the path. Round trip times are also displayed.

Part 5: Explore the nslookup Troubleshooting Tool

The **nslookup** tool used for querying Domain Name System (DNS) to obtain domain name to IP address mapping. This tool is useful to determine if the DNS server configured on a specific host is resolving hostnames to IP addresses.

Step 1: Query a domain.

To use nslookup, you need to type the hostname you are trying to resolve to an IP address. This will use the configured DNS server to find the IP address. You can also specify a DNS server to use.

Usage: nslookup [HOST] [SERVER]

a. Type **nslookup www.cisco.com** to determine the IP address of the domain.

```
devasc@labvm:~$ nslookup www.cisco.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
www.cisco.com canonical name = origin-www.cisco.com.
Name:   origin-www.cisco.com
Address: 173.37.145.84
Name:   origin-www.cisco.com
Address: 2001:420:1101:1::a

devasc@labvm:~$
```

The command returns the non-authoritative answer, and both the IPv4 and IPv6 name and address. The non-authoritative answer means that the server does not contain the original records of the domain's zone, rather, it is created from previous DNS lookups.

Note: Your output will most likely be different. However, you should see an IPv4 and IPv6 address.

Step 2: Query an IP address.

You can also look up IP addresses to discover the domain associated with it.

a. Query the DNS server for the IP address 8.8.8.8.

```
devasc@labvm:~$ nslookup 8.8.8.8
8.8.8.8.in-addr.arpa      name = dns.google.

Authoritative answers can be found from:

devasc@labvm:~$
```

Step 3: Query a domain using a specific DNS server.

a. Type **nslookup www.cisco.com 8.8.8.8** to determine the IP address of the domain according to Google's DNS.

```
devasc@labvm:~$ nslookup www.cisco.com 8.8.8.8
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
www.cisco.com canonical name = www.cisco.com.akadns.net.
www.cisco.com.akadns.net   canonical name = wwwds.cisco.com.edgekey.net.
wwwds.cisco.com.edgekey.net      canonical name =
wwwds.cisco.com.edgekey.net.globalredir.akadns.net.
wwwds.cisco.com.edgekey.net.globalredir.akadns.net   canonical name =
e2867.dsca.akamaiedge.net.
Name:  e2867.dsca.akamaiedge.net
Address: 23.205.37.210
Name:  e2867.dsca.akamaiedge.net
Address: 2600:1406:22:182::b33
Name:  e2867.dsca.akamaiedge.net
Address: 2600:1406:22:19c::b33

devasc@labvm:~$
```

Notice that by using this method, the server resolved the address to three different IP addresses, all different from the previous DNS query. These servers have a different cache of DNS queries to www.cisco.com.