# 12.7.4 Troubleshooting Tool Facts

When troubleshooting network communications, there are a number of considerations you have to take into account. There is also a number of tools specifically designed to troubleshoot network communications.

This lesson covers the following topics:

- Network communication troubleshooting
- Network communications troubleshooting tools

## Network Communication Troubleshooting

When experiencing problems with network communications, consider the following:

| Problem | Considerations |
|---|---|
| Physical issues | The best way to verify if a connection is valid is to check the link light on both the workstation and the switch. If the link light is unlit, try the following:<br>▪ Swap the cables. This will help determine whether the cable is the problem.<br>▪ Try using a different switch port for the connection.<br>▪ Make sure that the card is properly seated.<br>▪ Use loopback plugs to test network cards and cable testing devices to test network cables. |
| Interference | Interference is caused by electromagnetic fields or radio frequency interference. Check the following:<br>▪ For wired cables, make sure wires are not routed next to motors or fluorescent lights that can cause interference.<br>▪ For wireless devices, make sure there are no other devices in the area transmitting on the same frequency and channel (e.g., microwaves or cordless phones).<br>▪ Check to make sure that the cable is not kinked or worn. Cables should be routed through walls or ceilings, not strung across the floor. If a cable must run across the floor, encase the cable to prevent wear and secure the cable in place to prevent tripping accidents. Worn cables might introduce some interference, or simply prevent signals from being sent properly. |
| Network issues | If the device and its connection appear to be working correctly, check the following:<br>▪ Check firewalls on both end devices to see if communications are being blocked by a host-based firewall.<br>▪ Check the service on the target device to make sure that it is running and is properly configured. |
| Network performance | Network performance is comprised of many factors. A significant factor is *latency*, which is the amount of time it takes data to travel from one point of the network to another. To improve latency, look at:<br>▪ Increasing or improving *bandwidth* which is the size of the communication channel.<br>▪ Increasing or improving *throughput* which is the ability of the system to send and receive data.<br>▪ Eliminating *saturation* of bandwidth or throughput. Saturation indicates that maximum capacity has been achieved. Exceeding that point may result in a bottleneck.<br>Technologies that may improve latency are:<br>▪ Remote Direct Memory Access (RDMA) drivers. They provide high-throughput, low-latency communication that minimizes CPU usage. The drivers quickly transfer the contents of a memory buffer to a buffer on a remote system. RDMA drivers use one of the following communication standards:<br>  ▪ Infiniband (IB)<br>  ▪ Internet Wide Area RDMA Protocol (iWARP)<br>  ▪ RDMA over Converged Ethernet (RoCE)<br>▪ Unix sockets instead of localhost. You can improve latency by using a Unix domain socket for exchanging data between processes executing on the same host operating system. Unix socket communication occurs in the operating system kernel. The file system is used as the address name space and the two processes can communicate by opening the same socket. |

## Network Communications Troubleshooting Tools

The following table compares some of the tools for troubleshooting network communication problems:

| Tool | Function |
|---|---|
| **ping** | Verifies connectivity between hosts within the network.<br>▪ Ping a host using its IP address. If there is no response, try to ping another host. |

| | |
|---|---|
| | ▪ If your computer cannot communicate with any other computer, check the network cable, the network interface card, or the IP address configuration on your computer.<br>▪ If your computer can communicate with computers on the local network, but can't communicate with remote computers (such as the internet), verify the default gateway configuration on your computer.<br>▪ If all computers on the local network cannot communicate with any remote computer, troubleshoot the router's connection to the remote network.<br>▪ Ping a host using its DNS name. If a ping by IP address works, but a ping by DNS name fails, then there is probably a name resolution problem.<br>▪ Use the **-c** option to specify how many ICMP echo requests to send to the destination.<br><br>IPv6 communications can be tested using ping, but the **ping6** command must be used. |
| **netstat** | Displays a list of network connections (e.g., sockets), the routing table, and information about the network interface. A socket is an endpoint of a bidirectional communication flow across a computer network. Use the following options for additional information:<br><br>▪ **-a** lists both listening and non-listening ports.<br>▪ **-i** displays a table of all network interfaces.<br>▪ **-l** lists listening sockets.<br>▪ **-s** displays statistics for each protocol.<br>▪ **-r** displays the routing table, which includes the IP address of the default gateway.<br><br>The **netstat** command is being replaced by **ss**, **ip route** (for **netstat -r**), **ip -s link** (for **netstat -i**), and **ip maddr** (for **netstat -g**) |
| **nc**<br>**ncat** | Tests communications between network hosts. The netcat (**nc** or **ncat**) command establishes a TCP or UDP connection between two computers. The procedure for using nc is to:<br><br>▪ Open a listening TCP or UDP socket on one host. The syntax is **nc -l *port_number***. The **-l** option tells netcat to wait and listen for incoming connections. If no protocol is specified, then TCP is used by default. To use UDP instead of TCP, include the **-u** option in the command.<br>▪ Connect to the listening socket on the first host from another host. The syntax is **nc *ip_address port_number***.<br><br>After the connection is established, text entered at the prompt of the second computer should appear on the screen of the first computer.<br><br>You must open the appropriate ports in the host firewalls of both systems. |
| **traceroute**<br>**tracepath** | Tests connectivity between devices, show the path between the two devices. **traceroute:**<br><br>▪ Can help track down which router (known as a hop) in the route is not working correctly.<br>▪ Displays the Round Trip Time (RTT) for each hop. The RTT is the time difference between when the probe was sent from **traceroute** and the time the response arrived for each packet.<br><br>**tracepath** is similar to **traceroute**, but does not require super user privileges.<br><br>To test IPv6 routing, use the **traceroute6** or the **tracepath6** commands instead of traceroute. |
| **nslookup** | Sends a name resolution request. To use nslookup:<br><br>1. Enter **nslookup** at the shell prompt.<br>2. Enter the hostname or IP address, such as 192.168.1.1.<br>3. The DNS server should respond with the requested mapping.<br>4. Enter **exit** when finished.<br><br>The **nslookup** command is being replaced by the **host** and **dig** commands. |
| **dig** | Sends a name resolution request and receive extensive information about the hostname or IP address. Consider the following options:<br><br>▪ **a** resolves a record information.<br>▪ **ptr** resolves a ptr record.<br>▪ **cname** resolves cname record information.<br>▪ **p** queries a specific port on the host. |

| | |
|---|---|
| | ▪ **in** resolves internet record information.<br>▪ **mx** resolves mx record information.<br>▪ **soa** resolves start of authority information. |
| **ss** | Dumps socket statistics. Provides detailed information about communication with other hosts, networks, services, network connections, networking protocol statistics, and Linux socket connections. Consider the following options:<br>▪ **-a** displays all sockets.<br>▪ **-t** displays only TCP sockets.<br>▪ **-u** displays only UDP sockets.<br>▪ **-l** displays listening sockets.<br>▪ **-m** shows socket memory usage.<br>▪ **-p** shows process using socket.<br>▪ **ss > *ss_output*** sends the output to a file. |
| **nmcli** | Controls NetworkManager and get its status from the command line. Use **nmcli** as a complementary utility to *nm-applet* or other similar clients. Its main usage is on servers, headless machines, or for power users. Consider the following options:<br>▪ **-t** displays terse output. The output is suitable for scripts.<br>▪ **-p** displays pretty output that is easily readable by humans.<br>▪ **-m** specifies mode, tabular or multiline.<br>▪ **-f** specifies column names. |
| **nmtui** | Provides a text-base interface for controlling NetworkManager. Consider the following options:<br>▪ **-edit** displays a connection editor that supports adding, modifying, viewing, and deleting connections.<br>▪ **-connect** displays a list of available connections with options to activate or deactivate them.<br>▪ **-hostname** sets the system hostname. |
| **iftop** | Listens to network traffic on a named interface. If no interface is named it listens on the first interface that looks like an external interface. Be aware that **ifhop**:<br>▪ Displays a table of current bandwidth usage by pairs of hosts.<br>▪ Must be run with sufficient permissions to monitor all network traffic on the interface.<br>▪ Looks up the hostnames associated with addresses it finds in packets. This can cause substantial traffic. You can suppress display of DNS traffic by using filter code or switch it off with the **-n** option.<br>▪ Includes some of the following options:<br>  ▪ **-r** suppresses display of DNS traffic when the program is running.<br>  ▪ **-h** prints a summary of usage.<br>  ▪ **-N** suppresses resolving port number to service names.<br>  ▪ **-p** runs in promiscuous mode; counts traffic that does not pass directly through the specified interface.<br>  ▪ **-f** allows you to specify filters. |
| **iperf** | Performs network throughput measurements. To perform an **iperf** test, the user must establish both a server (to discard traffic) and a client (to generate traffic). Options include:<br>▪ **-f** specifies report format: [kmKM] Kbits, Mbits, KBytes, MBytes.<br>▪ **-i** pauses *n* seconds between periodic bandwidth reports.<br>▪ **-l** sets length read/write buffer (default 8 KB).<br>▪ **-o** specifies output file name for the report or error message.<br>▪ **-p** sets server port to listen on/connect to (default 5001)<br>▪ **-u** uses UDP rather than TCP |
| **tcpdump** | Dumps traffic on a network. Options include:<br>▪ **-A** prints each packet without the link level header in ASCII.<br>▪ **-B** sets the operating system capture buffer size.<br>▪ **-c** exits after receiving count packets.<br>▪ **-d** dumps the compiled packet-matching code in a human readable form to standard output and stops.<br>▪ **-dd** dumps packet-matching code as a C program fragment.<br>▪ **-D** prints the list of the network interfaces available on which tcpdump can capture packets. |
| **ipset** | Sets up, maintains, and inspects IP sets in the Linux kernel. Options include:<br>▪ **-n** creates a set identified with setname and specified type.<br>▪ **add** adds a given entry to the set.<br>▪ **del** deletes the specified entry from a set.<br>▪ **test** tests whether an entry is set or not.<br>▪ **-x** destroys specified set or all sets if no set is specified.<br>▪ **-t** lists the set names and header; suppresses listing set members.. |

| | |
|---|---|
| **mtr** | Combines the functionality of the **traceroute** and **ping** commands in a single network diagnostic tool. Options include:<br>▪ **-r** puts **mtr** into report mode, causing it to run for the number of cycles specified by the **-c** option, print statistics, and exit.<br>▪ **-t** forces **mtr** to use the curses based terminal interface if it is available.<br>▪ **-n** forces **mtr** to display numeric IP numbers and not try to resolve the host names.<br>▪ **-u** uses UDP datagrams instead of ICMP ECHO.<br>▪ **-4** uses IPv4 only.<br>▪ **-6** uses IPv6 only. |
| **arp** | Displays and modify the Internet-to-Ethernet address translation tables used by the Address Resolution Protocol, ARP. Options include:<br>▪ **-a** displays all of the current ARP entries.<br>▪ **-d** deletes the entry for the specified hostname. When combined with -a, deletes all entries and automatically disables hostname lookups.<br>▪ **-f** processes entries in the specified file to be set in the ARP tables.<br>▪ **-F** overwrites entries for a given host when used with **-f**.<br>▪ **-s** creates an ARP entry for the specified host and Ethernet address. This option is used with the **-f** option.<br>The **arp** command is being replaced by **ip n.** |
| **whois** | Looks up records in the databases maintained by several Network Information Centers (NICs) Options include:<br>▪ **-a** uses the American Registry for Internet Numbers (ARIN) database.<br>▪ **-d** uses the US Department of Defense database.<br>▪ **-g** uses the US non-military federal government database, which contains points of contact for subdomains of .GOV.<br>▪ **-h** uses the specified host instead of the default NIC. Either a host name or an IP address may be specified. |
| Wireshark<br>tshark | Tracks, intercepts, and logs network traffic. It can also generate a customized report from captured data. Use a CLI version of the Wireshark packet analyzer. Options include:<br>▪ **-i** specifies the name or index number of the interface.<br>▪ **-s** specifies the packet snapshot length.<br>▪ **-y** specifies the link type.<br>▪ **-c** stops capture after a specified number of packets.<br>▪ **-i** specifies a file to read from.<br>▪ **-i** specifies a file to output to.<br>▪ **-2** performs two-pass analysis. |