265-[NF]-Lab - Internet Protocol Troubleshooting Commands

Internet Protocol Troubleshooting Commands

Objectives

After completing this lab, you should be able to:

- Practice troubleshooting commands
- Identify how you can use these commands in customer scenarios

Duration

This lab requires approximately **30 minutes** to complete.

Scenario

You are a new network administrator who is troubleshooting customer issues.

AWS service restrictions

In this lab environment, access to AWS services and service actions might be restricted to the ones that you need to complete the lab instructions. You might encounter errors if you attempt to access other services or perform actions beyond the ones that this lab describes.

Accessing the AWS Management Console

1. At the top of these instructions, choose Start Lab to launch your lab.
   A **Start Lab** panel opens, and it displays the lab status.

   > **Tip**: If you need more time to complete the lab, choose the Start Lab button again to restart the timer for the environment.

2. Wait until you see the message *Lab status: ready*, then close the **Start Lab** panel by choosing the **X**.
3. At the top of these instructions, choose AWS.
   This opens the AWS Management Console in a new browser tab. The system will automatically log you in.

   > **Tip**: If a new browser tab does not open, a banner or icon is usually at the top of your browser with a message that your browser is preventing the site from opening pop-up windows. Choose the banner or icon and then choose **Allow pop ups**.

4. Arrange the AWS Management Console tab so that it displays along side these instructions. Ideally, you will be able to see both browser tabs at the same time so that you can follow the lab steps more easily.

Task 1: Use SSH to connect to an Amazon Linux EC2 instance

In this task, you will connect to a Amazon Linux EC2 instance. You will use an SSH utility to perform all of these operations. The following instructions vary slightly depending on whether you are using Windows or Mac/Linux.

Windows Users: Using SSH to Connect

💬 These instructions are specifically for Windows users. If you are using macOS or Linux, skip to the next section.

5. Select the Details drop-down menu above these instructions you are currently reading, and then select Show. A Credentials window will be presented.
6. Select the **Download PPK** button and save the **labsuser.ppk** file.
   Typically your browser will save it to the Downloads directory.
7. Make a note of the **PublicIP** address.
8. Then exit the Details panel by selecting the **X**.
9. Download **PuTTY** to SSH into the Amazon EC2 instance. If you do not have PuTTY installed on your computer, download it here.
10. Open **putty.exe**
11. Configure your PuTTY session by following the directions in the following link: Connect to your Linux instance using PuTTY
12. Windows Users: Select here to skip ahead to the next task.

Task 2: Practice troubleshooting commands

Recall

Some layers have commands related to them to help with troubleshooting. The following is an example of how the troubleshooting commands flow with the Open Systems Interconnection (OSI) model:

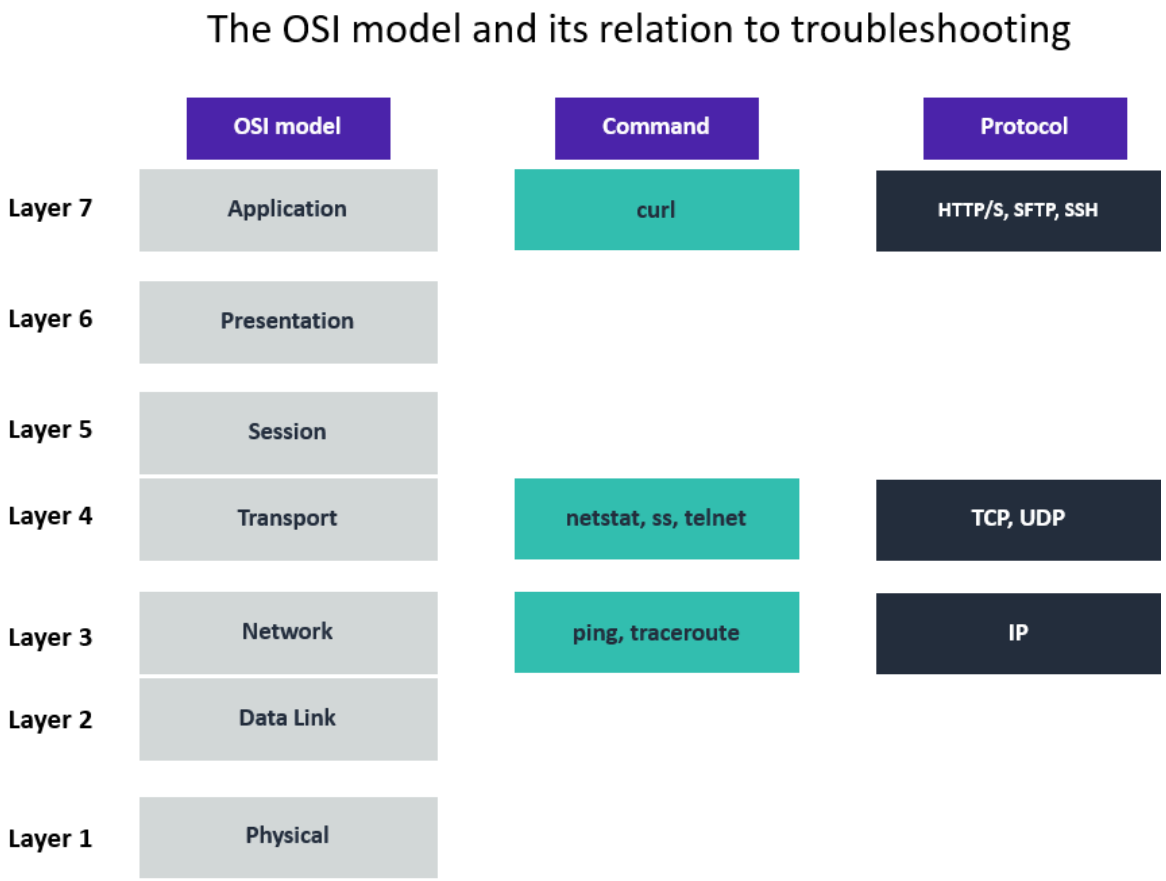

Figure: This is an example of how troubleshooting commands have similarities to the OSI model.

Layer 3 (network): The ping and traceroute commands

21. The following is an example of a customer scenario where you can use the **ping** command:

> The customer has launched an EC2 instance. To test connectivity to and from it, run the ping command. You can use this command to test connectivity and ensure that it allows Internet Control Message Protocol (ICMP) requests on the security level, such as security groups and network ACLs.

In the Linux terminal, run the following command, and press Enter:

```
ping 8.8.8.8 -c 5
```

This is the **ping** command. When you run this command, you can input an IP or URL followed by options. In this example, the **-c** stands for count, and **5** stands for how many requests you are requesting.

After you run this command, you should see a result similar to the following:

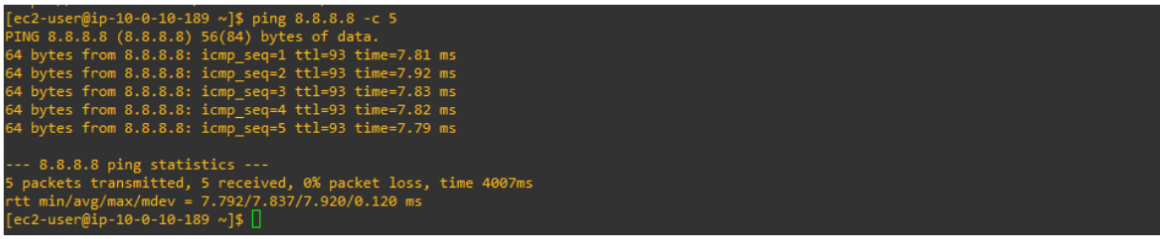

Figure: The ping command shows IP connectivity to a web server.

You can use the ping command for a few reasons, but the most common reason is to test connectivity to something such as a server. The ping command sends ICMP echo requests from your machine to the server that you are trying to reach (for example, amazon.com). The server sends an echo reply with a round-trip time. You use the ping command mostly to troubleshoot connectivity issues and reachability to a specific target. You can also use it to bring a specific network up if traffic needs to continuously flow through a network. You can also send a continuous ping.

22. The following is an example of a customer scenario where you can use the **traceroute** command:

> The customer is having latency issues. They say that their connection is taking a long time, and they are having packet loss. They aren't sure if it is related to AWS or their internet service provider (ISP). To investigate, you can run the traceroute command from their AWS resource to the server that they are trying to reach. If the loss happens toward the server, the issue is most likely the ISP. If the loss is toward AWS, you might need to investigate other factors that might limiting networking connectivity.

In the Linux terminal, run the following command, and press Enter:

```
traceroute 8.8.8.8
```

This is the **traceroute** command. You can input an IP or URL followed by options.

After you run this command, you should see a result similar to the following:

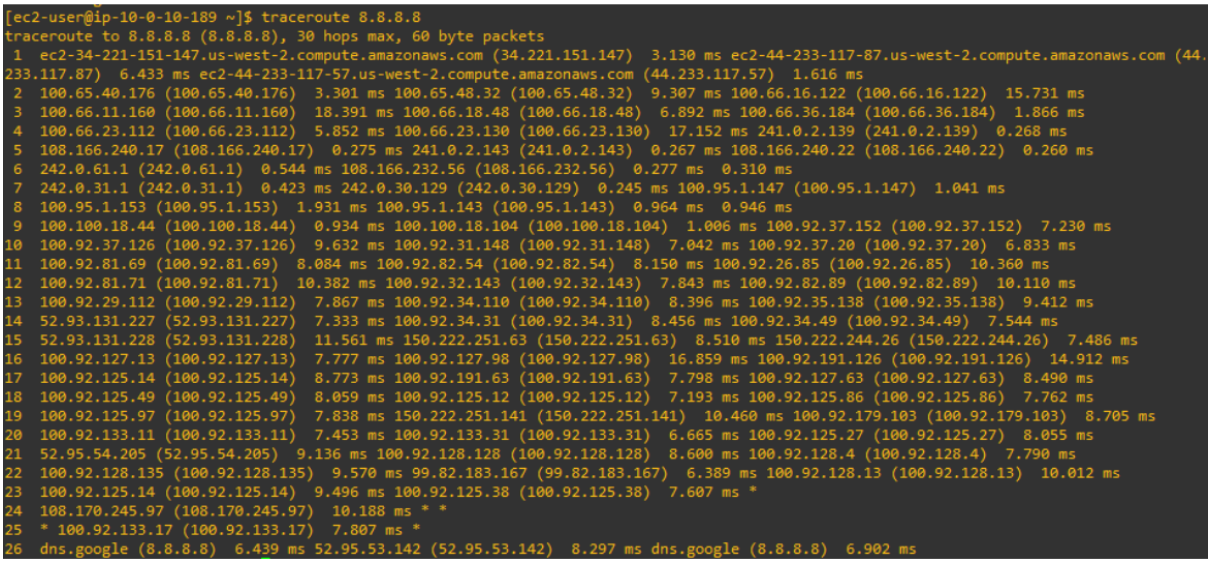

Figure: The traceroute command shows the path taken to the web server and the latency taken to it.

Packet loss, seen as percentages, can occur at each hop, and this loss usually occurs because of an issue with the user's local area network (LAN) or ISP

You can pinpoint an issue or error when the hostnames and IP addresses on either side of a jump have failed. Three asterisks (***) indicate a failed hop.

The traceroute command reports on the path and latency that the packet takes to get from your machine to the destination (8.8.8.8). Each server is called a hop. There can be packet loss, seen as percentages, at each loss, which is usually due to the user's local area network (LAN) or ISP; however, if the packet loss occurs toward the end of the route, then the issue is more than likely the server connection. You can pinpoint an issue or error when hostnames and IP addresses on either side of a failed jump, which looks like three asterisks (***).

Layer 4 (transport): The netstat and telnet commands

23. The following is an example of a customer scenario where you can use the **netstat** command:

    > Your company is running a routine security scan and found that one of the ports on a certain subnet is compromised. To confirm, you run the netstat command on a local host on that subnet to confirm if the port is listening when it shouldn't be.

    In the Linux terminal, run the following command, and press Enter:

    ```
    netstat -tp
    ```

    This is the **netstat** command. You can use the following options:

    - netstat -tp: Confirms established connections

- netstat -tlp: Outputs listening services
    - netstat -ntlp: Outputs listening services but does not resolve port numbers

    After you run this command, you should see a result similar to the following:

```
[ec2-user@ip-10-0-10-189 ~]$ netstat -tp
(No info could be read for "-p": geteuid()=1000 but you should be root.)
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0     64 ip-10-0-10-189.us-w:ssh 205.251.233.178:5477    ESTABLISHED -
```

Figure: The result of netstat -tp confirms established connections

The netstat command shows the current established TCP connections from which the host is listening. When troubleshooting networking issues starting with the host machine and working outward, you can run this command to understand which ports are listening and which are not. Because this command gives you a snapshot of your layer 4 connectivity, using this command will help you save time when trying to narrow down a large networking issue.

24. The following is an example of a customer scenario where you can use the **telnet** command:

    > The customer has a secure web server and has custom security group rules and network ACL rules configured. However, they are concerned that port 80 is open even though it shows their security settings indicate that their security group is blocking this port, you can run telnet 192.168.10.5 80 to ensure that the connection is refused.

    In the Linux terminal, run the following command, and press Enter to install telnet:

    ```
    sudo yum install telnet -y
    ```

    In the Linux terminal, run the following command, and press Enter:

    ```
    telnet www.google.com 80
    ```

    This is the **telnet** command. You can input an IP or URL followed by the port number to connect to that port.

    After you run this command, you should see a result similar to the following:

Figure: The telnet command confirms the TCP connection to a web server. It makes the HTTP request using telnet.

The telnet command confirms the TCP connection to a web server making an HTTP request if using port 80 to telnet. You can also use this command at layer 7. If you can successfully connect to the web server, then there is nothing blocking you or the server from connecting. If the connection fails with a message like "connection refused," then something is likely blocking the connection, such as a firewall or security group. If the connection fails with a message like "connection timed out," then then issue may be no network route or connectivity.

Layer 7 (application): The curl command

25. The following is an example of a customer scenario where you can use the **curl** command:

    > The customer has an Apache server running, and they want to test if they are getting a successful request (200 OK), which indicates that their website is running successfully. You run a curl request to see if the customer's Apache server returns a 200 OK.

    In the Linux terminal, run the following command, and press Enter:

    ```
    curl -vLo /dev/null https://aws.com
    ```

    This is the **curl** command. You can use the following command options:

    -I: This option provides header information and specifies that the request method is Head.
    -i: This option specifies that the request method is GET.
    -k: This option tells the command to ignore SSL errors.
    -v: This option is verbose. It shows what the computer is doing or what the software is loading during startup.
    -o /dev/null: This option will send HTML and CSS in response to null.

    After you run this command, you should see a result similar to the following:

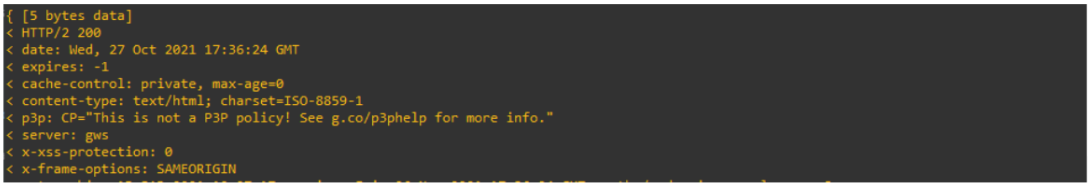

Figure: The results of the curl command: the output tests the connection to a web service, such as AWS, and submits the HTTP request.

You can use the curl command to transfer data between you and the server. The curl command can use many different protocols, but the most common are HTTP and HTTPS. You can use the curl command to troubleshoot communication from your local device to a server.

Lab Complete 🎓

🏁 Congratulations! You have completed the lab.

26. Select End Lab at the top of this page and then select Yes to confirm that you want to end the lab.
    A panel will appear, indicating that "DELETE has been initiated... You may close this message box now."
27. Select the **X** in the top right corner to close the panel.