



Lab - Comandos de solución de problemas del protocolo de Internet

Presentación realizada por Brendon Buriol,
Paulo Sena, Ignivé Amaro y Valeria Cantoni



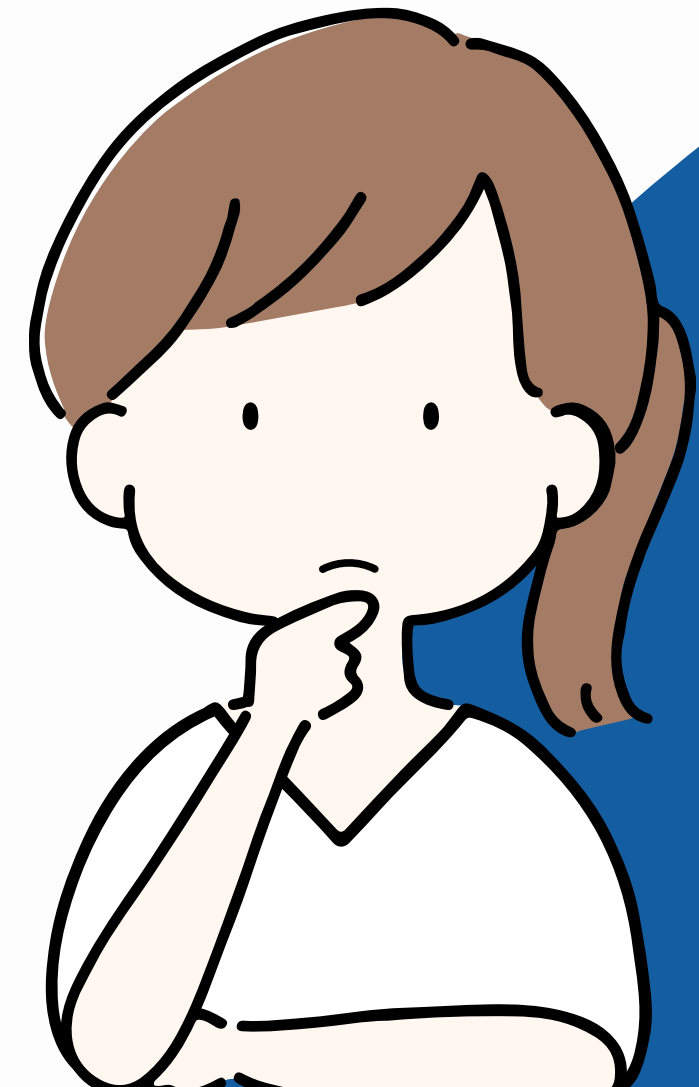


Objetivos

- Practicar los comandos de solución de problemas
- Identificar cómo puede usar estos comandos en las situaciones del cliente

Situación

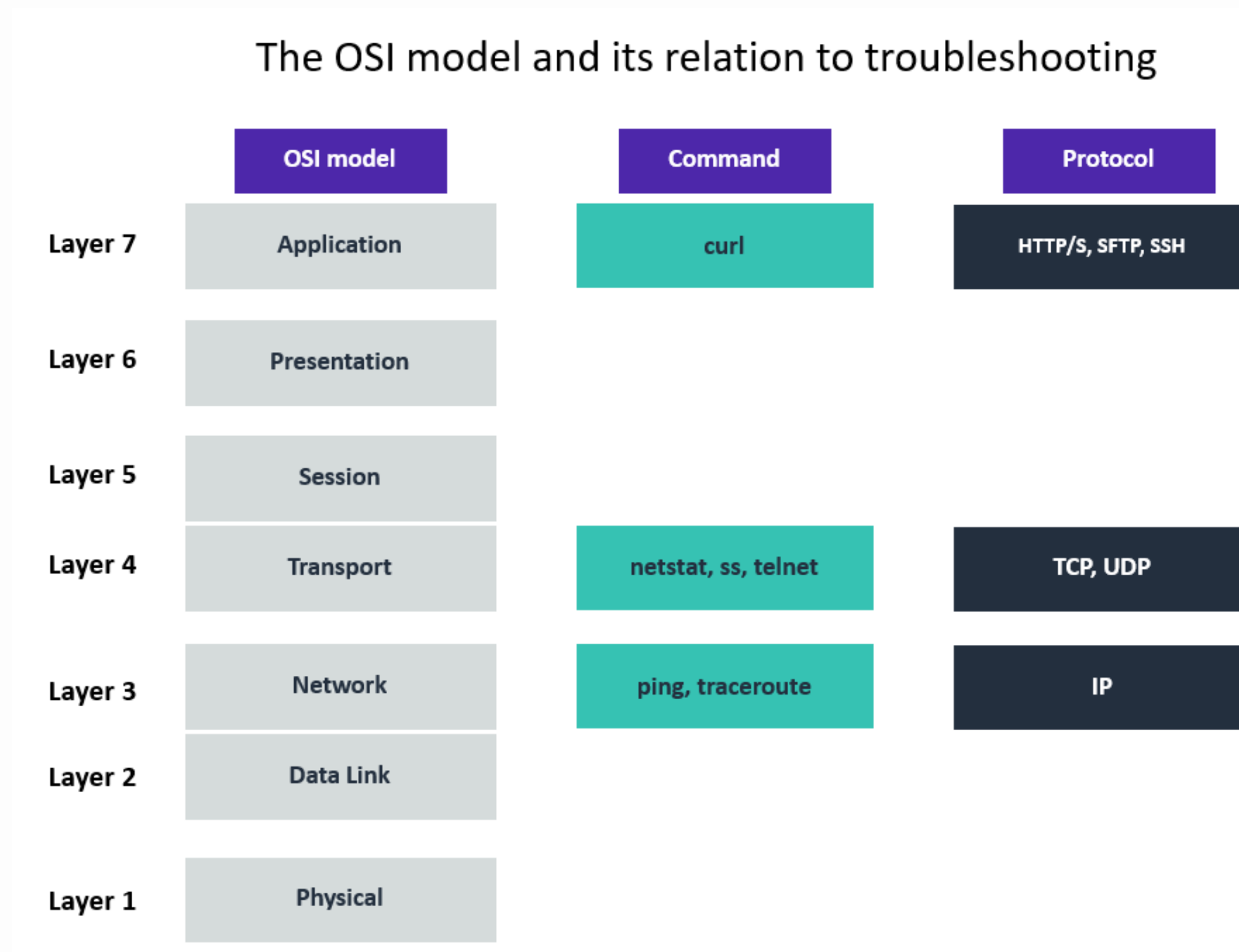
Es un administrador de red nuevo que está solucionando problemas de clientes.



Tarea 1: Practicar los comandos de solución de problemas

Recordatorio

Algunas capas tienen comandos relacionados con ellas para ayudar con la resolución de problemas. El siguiente es un ejemplo de cómo fluyen los comandos de solución de problemas con el modelo de interconexión de sistemas abiertos (OSI):



Capa 3 (red): los comandos ping y traceroute

El siguiente es un ejemplo de una situación de cliente en el que puede usar el comando ping:

El cliente ha lanzado una instancia EC2. Para probar la conectividad hacia y desde la instancia, ejecute el comando ping. Puede usar este comando para probar la conectividad y asegurarse de que permite las solicitudes del Protocolo de mensajes de control de Internet (ICMP) en el nivel de seguridad, como grupos de seguridad y ACL de red.

En la terminal de Linux, ejecute el siguiente comando y presione Enter: `ping 8.8.8.8 -c 5`

Este es el comando ping. Cuando ejecuta este comando, puede ingresar una IP o URL seguida de opciones. En este ejemplo, -c significa conteo y 5 representa cuántas solicitudes está solicitando.

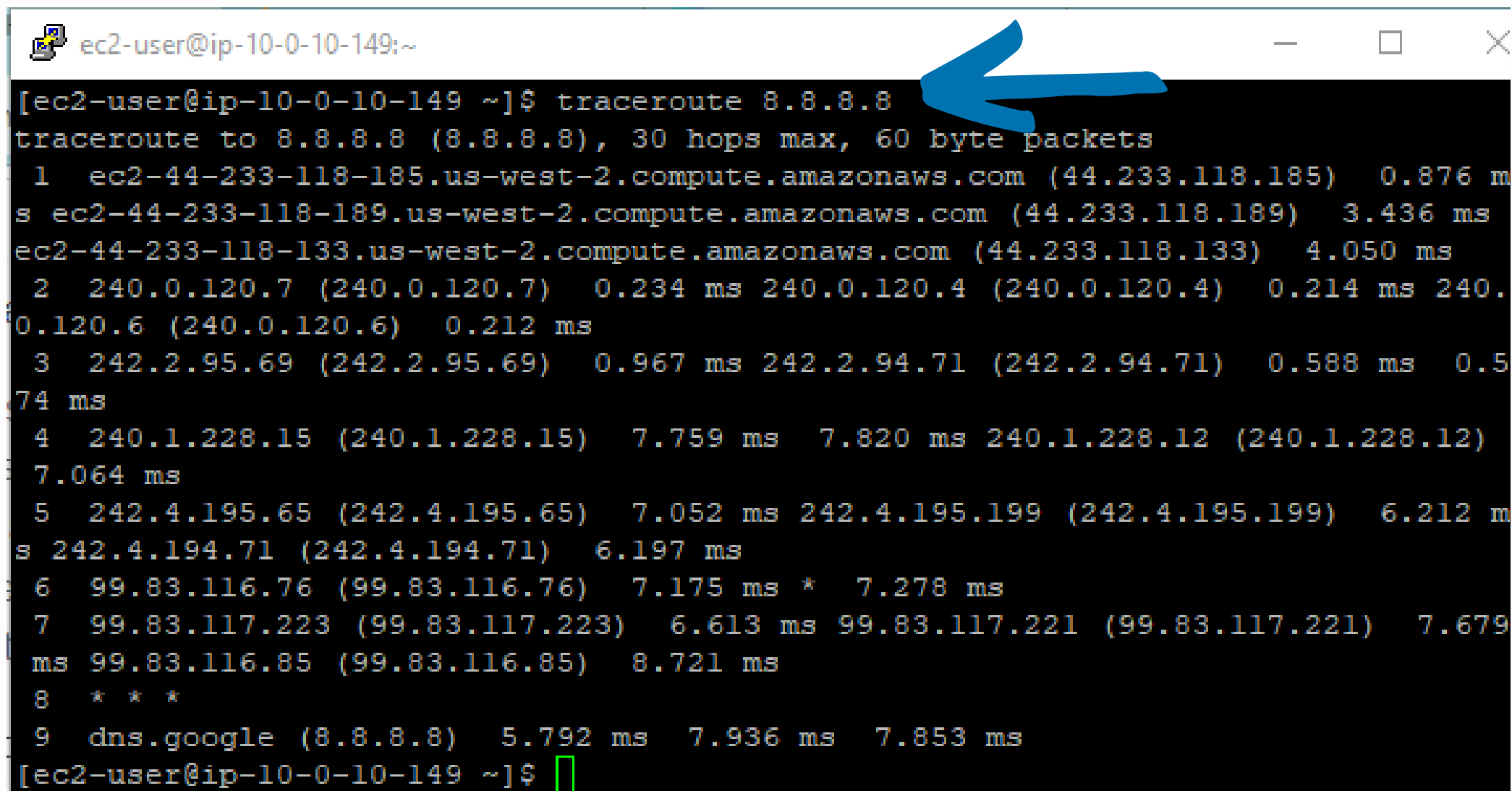
Después de ejecutar este comando, debería ver un resultado similar al siguiente:

```
ec2-user@ip-10-0-10-149:~  
Authenticating with public key "imported-openssh-key"  
#_#####  
~\##### Amazon Linux 2  
~~\#####  
~~\###| AL2 End of Life is 2025-06-30.  
~~\#/V~' '->  
~~~~_/ A newer version of Amazon Linux is available!  
~~._./ Amazon Linux 2023, GA and supported until 2028-03-15.  
_/_/ https://aws.amazon.com/linux/amazon-linux-2023/  
[ec2-user@ip-10-0-10-149 ~]$ ping 8.8.8.8 -c 5  
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=95 time=5.92 ms  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=95 time=5.95 ms  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=95 time=5.93 ms  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=95 time=5.93 ms  
64 bytes from 8.8.8.8: icmp_seq=5 ttl=95 time=6.00 ms  
  
--- 8.8.8.8 ping statistics ---  
5 packets transmitted, 5 received, 0% packet loss, time 4005ms  
rtt min/avg/max/mdev = 5.921/5.949/6.002/0.101 ms  
[ec2-user@ip-10-0-10-149 ~]$
```

El siguiente es un ejemplo de un escenario de cliente donde puede utilizar el comando traceroute :

El cliente tiene problemas de latencia. Dicen que su conexión está tardando mucho y que están perdiendo paquetes. No están seguros de si está relacionado con AWS o su proveedor de servicios de Internet (ISP). Para investigar, puede ejecutar el comando traceroute desde su recurso de AWS al servidor al que intentan acceder. Si la pérdida ocurre en el servidor, lo más probable es que el problema sea el ISP. Si la pérdida es para AWS, es posible que deba investigar otros factores que podrían limitar la conectividad de la red.

En la terminal de Linux, ejecute el siguiente comando y presione Entrar: `traceroute 8.8.8.8`



```
ec2-user@ip-10-0-10-149:~  
[ec2-user@ip-10-0-10-149 ~]$ traceroute 8.8.8.8  
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets  
 1  ec2-44-233-118-185.us-west-2.compute.amazonaws.com (44.233.118.185)  0.876 ms  
s ec2-44-233-118-189.us-west-2.compute.amazonaws.com (44.233.118.189)  3.436 ms  
ec2-44-233-118-133.us-west-2.compute.amazonaws.com (44.233.118.133)  4.050 ms  
 2  240.0.120.7 (240.0.120.7)  0.234 ms 240.0.120.4 (240.0.120.4)  0.214 ms 240.  
0.120.6 (240.0.120.6)  0.212 ms  
 3  242.2.95.69 (242.2.95.69)  0.967 ms 242.2.94.71 (242.2.94.71)  0.588 ms 0.5  
74 ms  
 4  240.1.228.15 (240.1.228.15)  7.759 ms 7.820 ms 240.1.228.12 (240.1.228.12)  
7.064 ms  
 5  242.4.195.65 (242.4.195.65)  7.052 ms 242.4.195.199 (242.4.195.199)  6.212 m  
s 242.4.194.71 (242.4.194.71)  6.197 ms  
 6  99.83.116.76 (99.83.116.76)  7.175 ms * 7.278 ms  
 7  99.83.117.223 (99.83.117.223)  6.613 ms 99.83.117.221 (99.83.117.221)  7.679  
ms 99.83.116.85 (99.83.116.85)  8.721 ms  
 8  * * *  
 9  dns.google (8.8.8.8)  5.792 ms 7.936 ms 7.853 ms  
[ec2-user@ip-10-0-10-149 ~]$
```

El comando traceroute informa sobre la ruta y la latencia que tarda el paquete en llegar desde su máquina al destino (8.8.8.8). Cada servidor se llama salto. Puede haber pérdida de paquetes, vista como porcentajes, en cada pérdida, que generalmente se debe a la red de área local (LAN) o ISP del usuario; sin embargo, si la pérdida de paquetes se produce hacia el final de la ruta, lo más probable es que el problema esté en la conexión del servidor.

Capa 4 (transporte): los comandos netstat y telnet

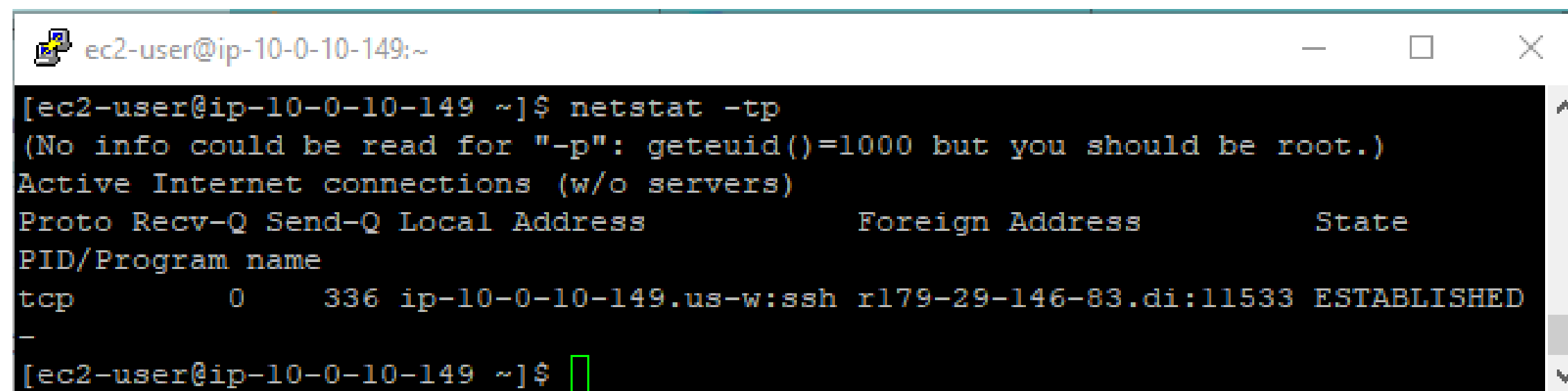
El siguiente es un ejemplo de una situación de cliente en el que puede usar el comando netstat:

Su empresa está ejecutando un análisis de seguridad de rutina y descubrió que uno de los puertos de una determinada subred está comprometido. Para confirmar, ejecute el comando netstat en un host local en esa subred para confirmar si el puerto está escuchando cuando no debería estarlo.

En la terminal de Linux, ejecute el siguiente comando y presione Enter: `netstat -tp`

Este es el comando netstat. Puede utilizar las siguientes opciones:

- netstat -tp: Confirma las conexiones establecidas.
- netstat -tp: Resultados de servicios de escucha.
 - netstat -tp: Genera servicios de escucha, pero no resuelve los números de puerto.
- Después de ejecutar este comando, debería ver un resultado similar al siguiente:



```
ec2-user@ip-10-0-10-149:~  
[ec2-user@ip-10-0-10-149 ~]$ 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      336 ip-10-0-10-149.us-w:ssh  rl79-29-146-83.di:11533 ESTABLISHED  
-  
[ec2-user@ip-10-0-10-149 ~]$
```

Ahora vamos a ver un ejemplo de un cliente donde podemos usar el comando telnet

El cliente tiene un servidor web seguro y tiene configuradas reglas de grupo de seguridad personalizadas y reglas de ACL de red. Sin embargo, les preocupa que el puerto 80 esté abierto a pesar de que muestra que su configuración de seguridad indica que su grupo de seguridad está bloqueando este puerto, puede ejecutar el comando telnet 192.168.10.5 80 para asegurarse de que se rechace la conexión.

En la terminal de Linux, ejecute el siguiente comando y presione Enter para instalar telnet:

```
sudo yum instalar telnet -y
```



En la terminal de Linux, ejecute el siguiente comando y presione Enter: Telnet www.google.com 80 para confirmar la conexión TCP a un servidor web

```
ec2-user@ip-10-0-10-149:~  
[ec2-user@ip-10-0-10-149 ~]$ telnet www.google.com 80  
Trying 142.251.33.68...  
Connected to www.google.com.  
Escape character is '^]'.  
█
```

```
ec2-user@ip-10-0-10-149:~  
[ec2-user@ip-10-0-10-149 ~]$ sudo yum install telnet -y  
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd  
amzn2-core | 3.6 kB 00:00  
Resolving Dependencies  
--> Running transaction check  
--> Package telnet.x86_64 1:0.17-65.amzn2 will be installed  
--> Finished Dependency Resolution  
  
Dependencies Resolved  
  
=====
```

Package	Arch	Version	Repository	Size
Installing:				
telnet	x86_64	1:0.17-65.amzn2	amzn2-core	64 k

```
=====
```

Transaction Summary

```
=====
```

Install 1 Package				
Total download size: 64 k				
Installed size: 109 k				
Downloading packages:				
telnet-0.17-65.amzn2.x86_64.rpm			64 kB	00:00
Running transaction check				
Running transaction test				
Transaction test succeeded				
Running transaction				
Installing :	1:telnet-0.17-65.amzn2.x86_64			1/1
Verifying :	1:telnet-0.17-65.amzn2.x86_64			1/1
Installed:				
telnet.x86_64 1:0.17-65.amzn2				
Complete!				
[ec2-user@ip-10-0-10-149 ~]\$ █				

Capa 7 (aplicación): el comando curl

El siguiente es un ejemplo de una situación del cliente en el que puede usar el comando curl:

El cliente tiene un servidor Apache ejecutándose y quiere probar si está recibiendo una solicitud exitosa (200 OK), lo que indica que su sitio web se está ejecutando correctamente. Puede ejecutar una solicitud del comando curl para ver si el servidor Apache del cliente devuelve una respuesta 200 OK.

En la terminal de Linux, ejecute el siguiente comando y presione Enter:

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

Este es el comando curl. Puede utilizar las siguientes opciones de comando:

- I: Esta opción proporciona información de encabezado y especifica que el método de solicitud es Head.
- i: Esta opción especifica que el método de solicitud es GET.
- k: Esta opción le dice al comando que ignore los errores de SSL.
- v: Estas opciones son detalladas. Muestra lo que está haciendo la computadora o lo que está cargando el software durante el inicio.
- o /dev/null: Esta opción enviará HTML y CSS en respuesta a nulo.

Después de ejecutar este comando, debería ver un resultado similar al siguiente:

```
{ [5 bytes data]
< HTTP/2 200
< content-type: text/html; charset=UTF-8
< server: Server
< date: Wed, 08 Nov 2023 23:40:25 GMT
< x-amz-rid: PGRYGPYQDSYE0V4Y7ZMG
< set-cookie: aws-priv=eyJ2IjoxLCJldSI6MCwic3QiOjB9; Version=1; Comment="Anonymous cookie for privacy regulations"; Domain=.aws.amazon.com; Max-Age=31536000; Expires=Thu, 07-Nov-2024 23:40:25 GMT; Path=/; Secure
< set-cookie: aws_lang=en; Domain=.amazon.com; Path=/
< x-frame-options: SAMEORIGIN
```

Resumen:

En esta práctica de laboratorio, se cubrieron los comandos de solución de problemas siguiendo la capa OSI. Las capas 3, 4 y 7 del modelo OSI se correlacionaron con comandos de solución de problemas y se dieron ejemplos en escenarios de clientes. Aunque esta no es una lista exhaustiva de comandos u opciones para solucionar problemas, estos son algunos de los comandos de solución de problemas más comunes que se utilizan para solucionar problemas comunes de red.