



## 265-[NF]-Lab -Comandos de solución de problemas del protocolo de Internet

### Datos Generales:

**Nombre:** Tomás Alfredo Villaseca Constantinescu

**País:** Chile

**Fecha:** 14/09/2023

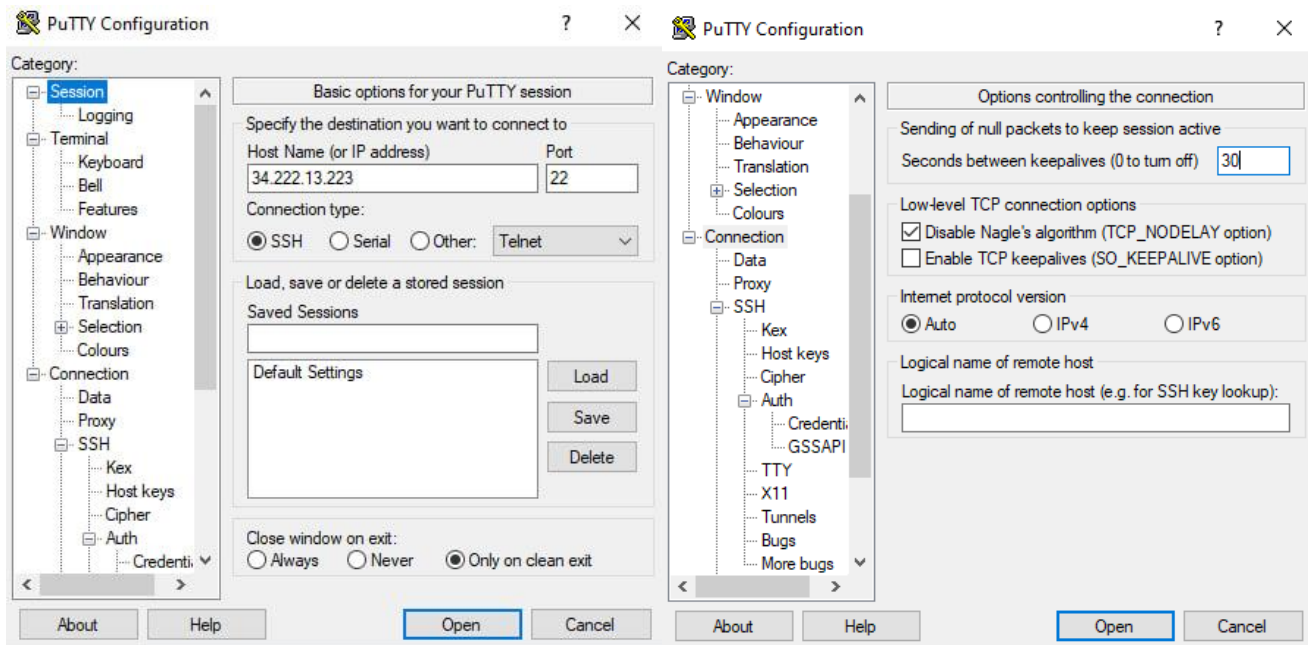
**Contacto:** [tomas.villaseca.c@gmail.com](mailto:tomas.villaseca.c@gmail.com)

Después de completar este módulo, podrá hacer lo siguiente:

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

# Tarea 1: conectarse a una instancia EC2 de Amazon Linux mediante SSH

1. Abrir Putty.exe: Se ingresa dirección IPv4 de la instancia EC2 en la sección Session.
2. En la sección Connection → SSH → Auth → Credentials se ingresa el archivo PPK descargado anteriormente.
3. En la sección Connection se establece **Seconds between keepalive en 30 (el valor predeterminado es 0).**



4. Se hace click en "Open" para validar y conectarse al Host.

```
ec2-user@ip-10-0-10-186:~  
login as: ec2-user  
Authenticating with public key "imported-openssh-key"  
  
  _ | _ | _ )  
  _ | ( _ /   Amazon Linux 2 AMI  
  _ | \ _ | _ |  
  
https://aws.amazon.com/amazon-linux-2/  
[ec2-user@ip-10-0-10-186 ~]$
```

## Tarea 2: Practicar los comandos de solución de problemas

Troubleshooting es un proceso sistemático para identificar y resolver problemas

Los pasos básicos del troubleshooting son los siguientes:

1. Identificar el problema
2. Recopilar información
3. Analizar la información
4. Implementar una solución
5. Probar la solución.

Similitud de los troubleshooting commands con el OSI Model:

### The OSI model and its relation to troubleshooting

	OSI model	Command	Protocol
Layer 7	Application	curl	HTTP/S, SFTP, SSH
Layer 6	Presentation		
Layer 5	Session		
Layer 4	Transport	netstat, ss, telnet	TCP, UDP
Layer 3	Network	ping, traceroute	IP
Layer 2	Data Link		
Layer 1	Physical		

## Situación

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

### Layer 3 (Network): Comandos ping y traceroute

**ping** = herramienta de diagnóstico de red que se utiliza para comprobar la conectividad entre dos dispositivos. Envía paquetes de datos a un dispositivo remoto y mide el tiempo que tardan en regresar.

- Funciona enviando un paquete de datos ICMP (Internet Control Message Protocol) al dispositivo remoto. El paquete ICMP contiene una solicitud de echo. El dispositivo remoto responde con un paquete de datos ICMP que contiene una respuesta de echo.
- También se puede utilizar para comprobar si un dispositivo remoto está disponible.
- -c = especifica el número de pings que se deben enviar (el predeterminado es 4).

#### Situación:

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.

 ec2-user@ip-10-0-10-12:~

```
[ec2-user@ip-10-0-10-12 ~]$ 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=99 time=7.92 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=99 time=7.96 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=99 time=7.92 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=99 time=8.26 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=99 time=8.38 ms

--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 7.925/8.093/8.389/0.217 ms
[ec2-user@ip-10-0-10-12 ~]$
```

**traceroute** = herramienta de diagnóstico de red que se utiliza para rastrear la ruta que sigue un paquete de datos desde su origen hasta su destino.

- Funciona enviando paquetes ICMP con un tiempo de vida (TTL) cada vez menor.
- TTL = valor que indica al dispositivo receptor cuántos saltos adicionales puede realizar el paquete antes de descartarlo.

A medida que el paquete viaja a través de la red, cada dispositivo de enrutamiento que lo recibe reduce el TTL en uno. Cuando el TTL llega a cero, el dispositivo de enrutamiento envía un mensaje de error ICMP al origen del paquete.

La salida del comando *traceroute* suele mostrar la siguiente información:

- El número de saltos que ha realizado el paquete
- La dirección IP de cada dispositivo de enrutamiento que ha recibido el paquete
- El tiempo que ha tardado el paquete en llegar a cada dispositivo de enrutamiento

El comando *traceroute* puede ser utilizado para solucionar problemas de conectividad de red. Por ejemplo, si un sitio web no está disponible, el comando *traceroute* puede utilizarse para determinar dónde se encuentra el problema.

## Situación:

El cliente tiene problemas de latencia. Dice que su conexión está tardando mucho y que están perdiendo paquetes. No está seguro de si está relacionado con AWS o con 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 hacia 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 pudieran limitar la conectividad de red.

```
ec2-user@ip-10-0-10-12:~$ traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  ec2-44-233-117-83.us-west-2.compute.amazonaws.com (44.233.117.83)  4.447 ms 244.5.0.203 (244.5.0.203)  43.682 ms ec2-34-221-1
 2  100.65.72.64 (100.65.72.64)  2.514 ms 100.65.74.144 (100.65.74.144)  7.640 ms 100.65.50.160 (100.65.50.160)  3.214 ms
 3  100.66.24.238 (100.66.24.238)  2.475 ms 108.166.232.80 (108.166.232.80)  0.270 ms 100.66.22.100 (100.66.22.100)  11.418 ms
 4  100.66.38.66 (100.66.38.66)  22.079 ms 241.0.1.142 (241.0.1.142)  0.243 ms 100.66.38.58 (100.66.38.58)  1.488 ms
 5  108.166.232.51 (108.166.232.51)  0.193 ms 108.166.232.57 (108.166.232.57)  0.204 ms 241.0.1.137 (241.0.1.137)  0.181 ms
 6  108.166.232.53 (108.166.232.53)  0.183 ms 108.166.232.58 (108.166.232.58)  0.183 ms 108.166.232.48 (108.166.232.48)  0.202 ms
 7  242.0.31.129 (242.0.31.129)  0.165 ms 108.166.232.60 (108.166.232.60)  32.301 ms 108.166.232.63 (108.166.232.63)  0.242 ms
 8  100.95.1.135 (100.95.1.135)  1.616 ms 100.95.17.133 (100.95.17.133)  0.971 ms 242.0.30.129 (242.0.30.129)  0.732 ms
 9  100.100.18.58 (100.100.18.58)  1.050 ms 100.100.18.44 (100.100.18.44)  0.305 ms 100.92.32.24 (100.92.32.24)  8.095 ms
10  100.92.37.154 (100.92.37.154)  6.691 ms 100.100.2.34 (100.100.2.34)  7.475 ms 100.92.31.168 (100.92.31.168)  7.204 ms
11  100.92.31.86 (100.92.31.86)  20.286 ms 100.92.32.58 (100.92.32.58)  16.014 ms 100.92.29.134 (100.92.29.134)  7.998 ms
12  100.92.32.28 (100.92.32.28)  7.542 ms 100.92.81.61 (100.92.81.61)  8.915 ms 100.92.26.37 (100.92.26.37)  10.987 ms
13  150.222.251.51 (150.222.251.51)  7.960 ms 100.92.35.116 (100.92.35.116)  15.856 ms 100.92.34.18 (100.92.34.18)  11.214 ms
14  100.92.28.86 (100.92.28.86)  7.409 ms 100.92.191.76 (100.92.191.76)  7.766 ms 100.92.35.16 (100.92.35.16)  8.319 ms
15  100.92.34.25 (100.92.34.25)  6.246 ms 100.92.29.55 (100.92.29.55)  9.008 ms 100.92.28.3 (100.92.28.3)  7.211 ms
16  100.92.191.52 (100.92.191.52)  9.732 ms 100.106.10.54 (100.106.10.54)  9.897 ms 100.91.185.44 (100.91.185.44)  11.597 ms
17  100.92.125.129 (100.92.125.129)  6.821 ms 6.502 ms 100.92.191.6 (100.92.191.6)  9.193 ms
18  100.92.127.133 (100.92.127.133)  8.369 ms 100.91.185.64 (100.91.185.64)  17.563 ms 100.92.127.109 (100.92.127.109)  8.001 ms
19  100.92.125.38 (100.92.125.38)  14.222 ms 100.92.125.108 (100.92.125.108)  9.411 ms 100.92.128.122 (100.92.128.122)  8.221 ms
20  100.92.133.41 (100.92.133.41)  24.672 ms 242.4.195.71 (242.4.195.71)  6.356 ms 100.92.133.77 (100.92.133.77)  6.166 ms
21  100.92.133.23 (100.92.133.23)  6.202 ms 52.93.131.19 (52.93.131.19)  9.575 ms 240.1.228.15 (240.1.228.15)  7.897 ms
22  100.92.125.123 (100.92.125.123)  12.250 ms 100.92.128.56 (100.92.128.56)  8.567 ms 242.4.195.69 (242.4.195.69)  6.228 ms
23  100.92.128.39 (100.92.128.39)  8.388 ms 100.92.128.85 (100.92.128.85)  8.495 ms 242.4.195.195 (242.4.195.195)  7.440 ms
24  100.92.125.129 (100.92.125.129)  7.429 ms 99.83.116.76 (99.83.116.76)  7.743 ms 100.92.125.84 (100.92.125.84)  12.916 ms
25  99.83.117.219 (99.83.117.219)  7.449 ms 100.92.133.57 (100.92.133.57)  13.507 ms *
26 * 240.1.228.14 (240.1.228.14)  9.057 ms dns.google (8.8.8.8)  6.531 ms
[ec2-user@ip-10-0-10-12 ~]$
```

## Layer 4 (Transport): Comandos netstat y telnet

**netstat** = herramienta de diagnóstico de red que se utiliza para mostrar información sobre las conexiones de red activas, los puertos abiertos y las tablas de enrutamiento.


- -n = muestra las direcciones IP y los números de puerto en forma numérica
- -l = muestra solo las conexiones TCP en estado de escucha
- -t = muestra las conexiones TCP en estado de escucha
- -p = especifica el protocolo que desea mostrar

La salida del comando netstat suele mostrar la siguiente información:


- El estado de la conexión
- La dirección IP de origen y destino
- El número de puerto de origen y destino
- El protocolo utilizado
- El identificador de proceso (PID) del proceso que inició la conexión

### Situación:

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

 ec2-user@ip-10-0-10-12:~

```
[ec2-user@ip-10-0-10-12 ~]$ netstat -tlnp
(No info could be read for "-p": geteuid()=1000 but you should be root.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 localhost:smtp          0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:sunrpc          0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN      -
tcp6       0      0 [::]:sunrpc             [::]:*                  LISTEN      -
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN      -
[ec2-user@ip-10-0-10-12 ~]$
```

 ec2-user@ip-10-0-10-12:~

```
[ec2-user@ip-10-0-10-12 ~]$ netstat -ntlnp
(No info could be read for "-p": geteuid()=1000 but you should be root.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:111            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22             0.0.0.0:*               LISTEN      -
tcp6       0      0 :::111                  :::*                    LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
[ec2-user@ip-10-0-10-12 ~]$
```



**telnet** = herramienta de línea de comandos que se utiliza para conectarse a un servidor remoto y acceder a una sesión de terminal.

### Situación:

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.

Usar comando `sudo yum install telnet -y` para instalar telnet:

```
ec2-user@ip-10-0-10-12:~  
[ec2-user@ip-10-0-10-12 ~]$ sudo yum install telnet -y  
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd  
amzn2-core  
Resolving Dependencies  
--> Running transaction check  
--> Package telnet.x86_64 1:0.17-65.amzn2 will be installed  
--> Finished Dependency Resolution  
  
Dependencies Resolved  
  
=====
```

Package	Arch
Installing:	
telnet	x86_64

```
=====
```

Transaction Summary

```
=====
```

Install 1 Package

Total download size: 64 k  
Installed size: 109 k  
Downloading packages:  
telnet-0.17-65.amzn2.x86\_64.rpm  
Running transaction check  
Running transaction test  
Transaction test succeeded  
Running transaction  
Installing : 1:telnet-0.17-65.amzn2.x86\_64  
Verifying : 1:telnet-0.17-65.amzn2.x86\_64  
  
Installed:  
telnet.x86\_64 1:0.17-65.amzn2  
  
Complete!  
[ec2-user@ip-10-0-10-12 ~]\$

Usar comando *telnet www.google.com 80* para verificar conexión TCP a un servidor web (puerto 80 = HTTP):

```
ec2-user@ip-10-0-10-12:~  
[ec2-user@ip-10-0-10-12 ~]$ telnet www.google.com 80  
Trying 142.251.33.100...  
Connected to www.google.com.  
Escape character is '^]'.  
█
```

Si la conexión falla con un mensaje como “conexión rechazada”, es probable que algo está bloqueando la conexión, como un firewall o un grupo de seguridad. Si la conexión falla con un mensaje como “tiempo de conexión agotado”, entonces el problema puede ser que no haya ruta de red o conectividad.

## Layer 7 (Application): Comando curl.

**curl** = herramienta de línea de comandos que se utiliza para transferir datos a través de varios protocolos, incluidos HTTP, HTTPS, FTP, SFTP, SCP y más.

- -I = proporciona información de encabezado y especifica que el método de solicitud es Head.
- -i = especifica que el método de solicitud es GET.
- -k = le dice al comando que ignore los errores de SSL.
- -v = 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.

El comando *curl* puede ser utilizado para una variedad de propósitos, incluyendo:

- Obtener datos de un servidor web
- Subir datos a un servidor web
- Transferir archivos entre sistemas
- Probar la conectividad de red

### Situación:

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.



Usar comando `curl -vLo /dev/null https://aws.com` para probar la conexión a un servicio web (ejemplo: aws.com) y envía la solicitud HTTP:

```
{ [5 bytes data]
< HTTP/2 200
< content-type: text/html; charset=UTF-8
< server: Server
< date: Fri, 15 Sep 2023 04:14:46 GMT
< x-amz-rid: 0Y50VGDZFAPRW7X3KTMM
< set-cookie: aws-priv=eyJ2IjoxLCJldSI6MCwic3QiOjB9; Version=1; Comment="Anonymous co
< set-cookie: aws_lang=en; Domain=.amazon.com; Path=/
< x-frame-options: SAMEORIGIN
< x-xss-protection: 1; mode=block
< strict-transport-security: max-age=63072000
< x-content-type-options: nosniff
< x-amz-id-1: 0Y50VGDZFAPRW7X3KTMM
< last-modified: Fri, 08 Sep 2023 09:28:50 GMT
< content-security-policy-report-only: default-src *; connect-src *; font-src * data
//prod-us-west-2.csp-report.marketing.aws.dev/submit
< vary: accept-encoding, Content-Type, Accept-Encoding, User-Agent
< x-cache: Miss from cloudfront
< via: 1.1 efe54e8b68e074d39b2ecd249f85100a.cloudfront.net (CloudFront)
< x-amz-cf-pop: HIO50-C1
< x-amz-cf-id:edrQl_A5is3VfFG9TlirVhENPGv24qLczMM7SlAeCGR9QLqImVXPNw==
<
{ [15491 bytes data]
100 282k 0 282k 0 0 1007k 0 --:--:-- --:--:-- --:--:-- 1007k
* Connection #1 to host aws.amazon.com left intact
[ec2-user@ip-10-0-10-12 ~]$
```

Se verifica que el sitio web aws.com se está ejecutando correctamente, lo anterior se comprueba dado que el servidor devuelve respuesta HTTP/2 200 (OK).

Laboratorio Completado

