
Ejercicios con Soluciones: LPIC-2 Objetivo 205.2 - Configuración Avanzada de Red y Resolución de Problemas

Escenario: Tienes dos servidores, `rocky` (Rocky Linux) y `ubuntu` (Ubuntu Server), ambos con el usuario `curso` con capacidad `sudo` y acceso `root`. Ambos están en la misma red y se pueden conectar por SSH. Tu objetivo es practicar la resolución de problemas y la monitorización de red.

Ejercicio 1: Manipulación de Rutas y Diagnóstico Básico

Objetivo: Simular un problema de enrutamiento y diagnosticarlo.

Parte A: Simular una Ruta Incorrecta (en `ubuntu`)

1. Conéctate a `ubuntu`:

```
ssh curso@ubuntu
```

2. Identifica la IP de `rocky`:

- Si no la sabes, puedes hacer `ping rocky` o verificar en `rocky` con `ip a`. Para este ejercicio, asumamos que la IP de `rocky` en tu red es `192.168.1.100` (ajusta según tu configuración real).
- La IP de `ubuntu` es `192.168.1.101` (ajusta si es diferente).

3. Añade una ruta incorrecta para la IP de `rocky` (¡temporalmente!):

- Esta ruta forzará el tráfico hacia `rocky` a pasar por un gateway incorrecto, interrumpiendo la comunicación directa.

```
sudo ip route add 192.168.1.100 via 192.168.1.250 dev enp0s3 # Asegúrate que 192.168.1.250 no existe en tu red
```

- Nota: `enp0s3` es el nombre de interfaz común en Ubuntu. Ajústalo si el tuyo es diferente (e.g., `eth0`).

4. Verifica la tabla de enrutamiento en `ubuntu`:

```
ip r
```

- Salida Esperada: Deberías ver la nueva ruta incorrecta para `192.168.1.100` apuntando a `192.168.1.250`.

5. Intenta hacer `ping` a `rocky` desde `ubuntu`:

```
ping -c 3 192.168.1.100
```

- Salida Esperada: Deberías ver que los pings fallan (e.g., "Destination Host Unreachable" o "Request timed out").

Parte B: Diagnóstico y Corrección

1. Desde `ubuntu`, usa `traceroute` para diagnosticar el problema:

```
traceroute 192.168.1.100
```

- Salida Esperada: `traceroute` mostrará que el tráfico intenta ir a `192.168.1.250` y falla, confirmando que la ruta incorrecta es el problema.

2. Elimina la ruta incorrecta en `ubuntu`:

```
sudo ip route del 192.168.1.100 via 192.168.1.250 dev enp0s3
```

3. Verifica la tabla de enrutamiento nuevamente:

```
ip r
```

- Salida Esperada: La ruta incorrecta ya no debería aparecer.

4. Intenta hacer `ping` a `rocky` de nuevo:

```
ping -c 3 192.168.1.100
```

- Salida Esperada: Los pings deberían funcionar ahora, demostrando que el problema de enrutamiento se ha resuelto.

Ejercicio 2: Monitoreo de Conexiones y Uso de Puertos

Objetivo: Identificar qué servicios están escuchando en qué puertos y qué conexiones activas existen.

Parte A: Listar Puertos en Escucha y Conexiones Activas

1. En `rocky` (Rocky Linux):

- Lista todos los puertos TCP y UDP en escucha, mostrando el proceso que los usa:

```
sudo ss -tulnp
```

- Salida Esperada (ejemplo): Verás líneas como `tcp LISTEN 0 128 0.0.0.0:22 0.0.0.0:* users:(("sshd",pid=XXX,fd=Y))`, indicando que `sshd` está escuchando en

el puerto 22.

2. Desde `ubuntu`, establece una conexión SSH a `rocky`:

```
ssh curso@rocky
# Ingresa tu contraseña. No cierres esta conexión.
```

3. En una nueva terminal en `rocky`, lista las conexiones TCP establecidas:

```
ss -tnap | grep ':22' # -t para TCP, -n para numérica, -a para todas, -p para proceso
```

- Salida Esperada: Deberías ver al menos dos líneas relacionadas con el puerto 22:
- Una en estado `LISTEN` (el servidor SSH escuchando).
- Otra en estado `ESTAB` (tu conexión SSH activa desde `ubuntu`).

Parte B: Identificar Procesos por Puerto

1. Desde la primera terminal en `ubuntu`, cierra la sesión SSH a `rocky`:

```
exit
```

2. En `rocky`, simula un servicio escuchando en un puerto personalizado:

- Abre una nueva terminal en `rocky` o regresa a la anterior.
- Usa `netcat` para escuchar en el puerto `12345`:

```
nc -lvp 12345
```

Esta terminal quedará bloqueada escuchando.

3. En otra terminal en `rocky` (o una nueva conexión SSH), usa `lsof` para encontrar el proceso que usa el puerto `12345`:

```
sudo lsof -i :12345
```

- Salida Esperada: Deberías ver el comando `nc` y su PID usando el puerto `12345`.

```
COMMAND   PID  USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
nc       XXXX curso    3u  IPv4  YYYYY      0t0  TCP *:12345 (LISTEN)
```

4. Desde `ubuntu`, intenta conectar al puerto `12345` de `rocky`:

```
nc -zv 192.168.1.100 12345
```

- Salida Esperada: Deberías ver "Connection to 192.168.1.100 12345 port [tcp/*] succeeded!"
- En la terminal de `rocky` donde `nc` estaba escuchando, deberías ver "Connection received on 192.168.1.101 4XXXXX" (la IP y puerto de origen de `ubuntu`).

5. En la terminal de `rocky` donde `nc` está escuchando, presiona `Ctrl+C` para detenerlo.

Ejercicio 3: Captura y Análisis de Tráfico con `tcpdump`

Objetivo: Observar el tráfico de red en vivo para diagnosticar problemas o verificar la comunicación.

Parte A: Capturar Tráfico ICMP (Ping)

1. En `rocky` (Rocky Linux), inicia `tcpdump` para capturar tráfico ICMP en tu interfaz principal:

```
sudo tcpdump -ni eth0 icmp # Reemplaza eth0 con tu interfaz real
```

- `n`: No resuelve nombres de host (muestra IPs numéricas).
- `i eth0`: Captura en la interfaz `eth0`.
- `icmp`: Filtra solo paquetes ICMP.
- Esta terminal quedará capturando.

2. Desde `ubuntu` (en otra terminal), haz `ping` a `rocky`:

```
ping -c 3 192.168.1.100
```

3. Observa la salida de `tcpdump` en `rocky`:

- Salida Esperada: Verás líneas que muestran los paquetes ICMP `echo request` (de `ubuntu` a `rocky`) y `echo reply` (de `rocky` a `ubuntu`), confirmando que el tráfico ICMP está fluyendo.

```
HH:MM:SS.ms IP 192.168.1.101 > 192.168.1.100: ICMP echo request, id 1234, seq 1,
length 64
HH:MM:SS.ms IP 192.168.1.100 > 192.168.1.101: ICMP echo reply, id 1234, seq 1,
length 64
```

4. En la terminal de `rocky` con `tcpdump`, presiona `Ctrl+C` para detener la captura.

Parte B: Capturar Tráfico SSH

1. En `rocky`, inicia `tcpdump` para capturar tráfico en el puerto SSH (22):

```
sudo tcpdump -ni eth0 port 22
```

- Esta terminal quedará capturando.

2. Desde `ubuntu`, establece una nueva conexión SSH a `rocky`:

```
ssh curso@rocky
```

3. Observa la salida de `tcpdump` en `rocky`:

- Salida Esperada: Verás los paquetes TCP correspondientes al establecimiento de la conexión SSH (SYN, SYN-ACK, ACK, etc.) y luego el tráfico de datos cifrados.

```
HH:MM:SS.ms IP 192.168.1.101.X > 192.168.1.100.22: Flags [S], seq YYYYY, win
ZZZZ, options [mss ..., sackOK,TS val ...,nop,wscale ...], length 0
HH:MM:SS.ms IP 192.168.1.100.22 > 192.168.1.101.X: Flags [S.], seq AAAAA, ack
BBBBB, win CCCCC, options [mss ..., sackOK,TS val ...,nop,wscale ...], length 0
... (y más tráfico cifrado)
```

4. Cierra la conexión SSH desde `ubuntu` (`exit`).

5. En la terminal de `rocky` con `tcpdump`, presiona `Ctrl+C` para detener la captura.

Ejercicio 4: Escaneo de Puertos con `nmap`

Objetivo: Identificar qué puertos están abiertos en un sistema remoto.

1. Desde `ubuntu`, escanea los puertos más comunes en `rocky` (`192.168.1.100`):

```
nmap 192.168.1.100
```

- Salida Esperada (ejemplo):

```
Starting Nmap 7.XX ( https://nmap.org ) at 2024-XX-XX HH:MM CEST
Nmap scan report for rocky (192.168.1.100)
Host is up (0.000XXs latency).
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
```

Esto te mostrará los puertos abiertos que `nmap` pudo detectar, como SSH (22) y posiblemente HTTP (80) si tienes un servidor web.

2. Desde `ubuntu`, realiza un escaneo de puertos específico para verificar si un puerto particular está abierto (e.g., puerto 12345):

- En `rocky`, abre el puerto `12345` temporalmente con `nc` (sin cerrar la terminal):

```
nc -lvp 12345
```

- Desde `ubuntu`, escanea solo el puerto `12345` en `rocky`:

```
nmap -p 12345 192.168.1.100
```

- Salida Esperada:

  ```
  PORT      STATE SERVICE
  12345/tcp open  netc
  ```

  Esto confirma que el puerto `12345` está abierto en `rocky`.

- En `rocky`, presiona `Ctrl+C` en la terminal donde `nc` está escuchando para cerrar el puerto.
- Desde `ubuntu`, escanea de nuevo el puerto `12345`:

  ```
  nmap -p 12345 192.168.1.100
  ```

- Salida Esperada:

  ```
  PORT      STATE SERVICE
  12345/tcp closed netc
  ```

  Esto confirma que el puerto `12345` ahora está cerrado.

Estos ejercicios prácticos te permitirán familiarizarte con las herramientas de diagnóstico y resolución de problemas de red en un entorno Linux, lo cual es fundamental para el objetivo 205.2.