
Documento de Estudio: LPIC-2 Objetivo 205.2 - Configuración Avanzada de Red y Resolución de Problemas

Peso del Objetivo: 3

Descripción General

El objetivo 205.2 va más allá de la configuración básica, adentrándose en la resolución de problemas de red y la monitorización avanzada. Un administrador de sistemas Linux debe ser capaz de diagnosticar y solucionar fallos de conectividad, analizar el tráfico, y comprender cómo las diferentes capas de la red interactúan para identificar la raíz de un problema.

Áreas de Conocimiento Clave Desarrolladas

1. Utilidades para Manipular Tablas de Enrutamiento

La tabla de enrutamiento es el "mapa" que usa el sistema operativo para decidir por dónde enviar los paquetes de datos. Entender y manipular esta tabla es fundamental para asegurar la conectividad entre diferentes subredes.

- Comprendiendo el Enrutamiento: Cuando un sistema necesita enviar un paquete a una dirección IP, primero consulta su tabla de enrutamiento. Si la dirección de destino está en la misma subred que alguna de sus interfaces, el paquete se envía directamente. Si no, busca una ruta que le indique a qué gateway (puerta de enlace) debe enviar el paquete para que este lo reenvíe hacia su destino final. La ruta por defecto (o default gateway) es la ruta que se usa cuando no hay una ruta más específica para un destino.
- Manipulación de Rutas con `ip route` (Preferida):
- Mostrar la tabla de enrutamiento:

```
ip route show
# o la forma corta
ip r
```

Ejemplo de Salida:

```
default via 192.168.1.1 dev eth0 proto dhcp metric 100
10.0.0.0/8 via 192.168.1.254 dev eth0
192.168.1.0/24 dev eth0 proto kernel scope link src 192.168.1.100 metric 100
```

- `default via 192.168.1.1 dev eth0`: La ruta por defecto, todo el tráfico no específico va a 192.168.1.1 a través de `eth0`.
- `10.0.0.0/8 via 192.168.1.254 dev eth0`: Una ruta estática para la red 10.0.0.0/8, se accede a ella a través del gateway 192.168.1.254.

- Añadir una ruta estática (IPv4): Para acceder a la red `172.16.0.0/24` a través de un router en `192.168.1.254`:

```
sudo ip route add 172.16.0.0/24 via 192.168.1.254 dev eth0
```

- Añadir una ruta estática (IPv6): Para la red `2001:db8:2::/64` a través de un gateway en `2001:db8:1::1`:

```
sudo ip -6 route add 2001:db8:2::/64 via 2001:db8:1::1 dev eth0
```

- Añadir o cambiar la ruta por defecto (gateway):

```
sudo ip route add default via 192.168.1.1 dev eth0
```

  Nota: Si ya existe una ruta por defecto, primero deberías eliminarla o usar `replace`.

- Eliminar una ruta estática:

```
sudo ip route del 172.16.0.0/24 via 192.168.1.254 dev eth0
```

- Eliminar la ruta por defecto:

```
sudo ip route del default
```

- Manipulación de Rutas con `route` (Legado):
- Mostrar la tabla de enrutamiento (numérica):

```
route -n
```

  Ejemplo de Salida:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

- Añadir una ruta estática:

```
sudo route add -net 172.16.0.0 netmask 255.255.255.0 gw 192.168.1.254 dev eth0
```

- Eliminar una ruta estática:

```
sudo route del -net 172.16.0.0 netmask 255.255.255.0 gw 192.168.1.254
```

2. Utilidades para Configurar y Manipular Interfaces de Red Ethernet

Más allá de la asignación básica de IP, el control del estado y las propiedades de las interfaces es crucial.

- `ip link` (Preferida):
- Habilitar/Deshabilitar una interfaz:

```
sudo ip link set eth0 up   # Habilitar
sudo ip link set eth0 down # Deshabilitar
```

- Configurar la MTU (Maximum Transmission Unit): La MTU define el tamaño máximo de paquete que una interfaz puede enviar sin fragmentar. Un MTU incorrecto puede causar problemas de rendimiento o conectividad.

```
sudo ip link set eth0 mtu 1500
```

- Activar/Desactivar el modo promiscuo: Permite a una interfaz capturar todo el tráfico que ve, no solo el dirigido a ella. Útil para herramientas de análisis como `tcpdump`.

```
sudo ip link set eth0 promisc on
sudo ip link set eth0 promisc off
```

- `ifconfig` (Legado):
- Habilitar/Deshabilitar una interfaz:

```
sudo ifconfig eth0 up
sudo ifconfig eth0 down
```

- Configurar la MTU:

```
sudo ifconfig eth0 mtu 1500
```

3. Utilidades para Manipular el Estado de los Dispositivos de Red

Monitorizar el estado operativo de las interfaces es el primer paso en el diagnóstico.

- Verificar el estado de la interfaz:
- `ip link show`: Muestra si la interfaz está `UP` (arriba) o `DOWN` (abajo), el estado del `LINK` (conectado físicamente) y otros detalles.

```
ip link show eth0
# Ejemplo de salida:
# 2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
mode DEFAULT group default qlen 1000
```

```
#     link/ether 08:00:27:xx:xx:xx brd ff:ff:ff:ff:ff:ff
```

`UP,LOWER_UP` indica que la interfaz está habilitada y físicamente conectada.

- Estadísticas de tráfico:
- `ip -s link show eth0`: Muestra estadísticas detalladas de paquetes transmitidos (TX) y recibidos (RX), incluyendo errores y drops.

```
ip -s link show eth0
# Ejemplo de salida (fragmento):
#     RX: bytes  packets  errors  dropped overrun mcast
#     12345678   98765    0       0       0       0
#     TX: bytes  packets  errors  dropped carrier collsns
#     87654321   54321    0       0       0       0
```

Esto es útil para ver si hay problemas a nivel de capa de enlace (errores, paquetes caídos).

4. Utilidades para Monitorizar y Analizar el Tráfico TCP/IP

Estas herramientas son esenciales para el diagnóstico profundo, permitiéndote ver qué conexiones existen, qué puertos están abiertos y qué datos están fluyendo.

- `ss` (Socket Statistics - Preferida): Muestra información sobre sockets (conexiones de red). Es más rápido y ofrece más características que `netstat`.
- Listar todos los sockets TCP en escucha (listening) y conexiones establecidas (numerical, no resuelve nombres ni servicios, con nombre de programa/PID):

```
ss -tulnp
```

Ejemplo de Salida:

```
Netid  State      Recv-Q Send-Q Local Address:Port    Peer Address:Port
tcp    LISTEN     0      128    0.0.0.0:22            0.0.0.0:* users:
(("sshd",pid=1000,fd=3))
tcp    ESTAB      0      0      192.168.1.100:22      192.168.1.1:54321 users:
(("sshd",pid=1001,fd=3))
udp    UNCONN     0      0      0.0.0.0:68            0.0.0.0:* users:
(("dhclient",pid=1002,fd=6))
```

- Resumen de estadísticas de sockets:

```
ss -s
```

Ejemplo de Salida:

```
Total: 1234 (kernel 1235)
TCP:   3 (estab 1, closed 0, orphaned 0, synrecv 0, timewait 0/0), ports 0
```

Muestra un resumen rápido de las conexiones activas.

- `netstat` (Network Statistics - Legado): Proporciona estadísticas de red.

- Listar sockets TCP y UDP en escucha con PID y nombre de programa:

```
sudo netstat -tulnp
```

  Nota: La salida es similar a `ss -tulnp`.

- Mostrar la tabla de enrutamiento:

```
netstat -r
```

  Nota: La salida es similar a `route -n`.

- `lsof` (List Open Files): Puede listar procesos que tienen archivos abiertos, incluyendo sockets de red. Es invaluable para saber qué proceso está utilizando un puerto específico.
- Ver qué proceso usa el puerto 80 (HTTP):

```
sudo lsof -i :80
```

  Ejemplo de Salida:

```
COMMAND     PID  USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
apache2    1234 apache    4u  IPv6  56789      0t0  TCP *:http (LISTEN)
```

- Ver conexiones de un host remoto específico:

```
sudo lsof -i @192.168.1.5
```

- `ping`, `ping6`: Herramientas básicas pero esenciales para probar la conectividad a nivel de red (Capa 3). Envían paquetes ICMP/ICMPv6.
- Probar conectividad IPv4:

```
ping google.com       # Por nombre de host
ping 8.8.8.8          # Por dirección IP
```

- Probar conectividad IPv6:

```
ping6 ipv6.google.com
ping6 2001:4860:4860::8888
```

- Opciones útiles:
- `-c <count>`: Enviar un número específico de paquetes. `ping -c 4 192.168.1.1`
- `-I <interface>`: Especificar la interfaz de origen. `ping -I eth0 8.8.8.8`
- `nc` (netcat): La "navaja suiza" de las redes. Puede abrir conexiones TCP/UDP, escuchar puertos, y

transferir datos. Muy útil para pruebas rápidas de conectividad a puertos específicos.

- Probar si un puerto TCP está abierto en un servidor remoto:

```
nc -zv www.example.com 80 # Ver si el puerto 80 está abierto
nc -zv 192.168.1.100 22   # Ver si SSH está escuchando
```

  Salida esperada si está abierto: `Connection to 192.168.1.100 22 port [tcp/ssh] succeeded!`

- Abrir un puerto para escuchar conexiones (servidor):

```
nc -lvp 12345
```

  Nota: Para detener, presiona `Ctrl+C`.

- Conectarse a un puerto que está escuchando (cliente):

```
nc 192.168.1.100 12345
```

  Una vez conectados, lo que escribas en una terminal aparecerá en la otra.

- `tcpdump`: Una herramienta extremadamente potente para la captura y análisis de paquetes de red. Requiere un buen conocimiento de las opciones de filtrado.
- Capturar todo el tráfico en una interfaz:

```
sudo tcpdump -i eth0
```

- Capturar solo tráfico SSH (puerto 22) en una interfaz:

```
sudo tcpdump -i eth0 port 22
```

- Capturar tráfico hacia/desde un host específico:

```
sudo tcpdump -i eth0 host 192.168.1.5
```

- Capturar solo paquetes ICMP (ping):

```
sudo tcpdump -i eth0 icmp
```

- No resolver nombres ni servicios (muestra IPs y números de puerto):

```
sudo tcpdump -ni eth0
```

- Guardar la captura en un archivo para análisis posterior con Wireshark:

  ```
  sudo tcpdump -i eth0 -w capture.pcap
  ```

  Para ver el contenido: `tcpdump -r capture.pcap` o abrir con Wireshark.

- `nmap` (Network Mapper): Herramienta robusta para escanear redes, descubrir hosts, escanear puertos, detectar servicios y sistemas operativos. Aunque se asocia a la seguridad, es una herramienta fundamental de diagnóstico de red.
- Escaneo básico de puertos (los 1000 más comunes) de un host:

  ```
  nmap 192.168.1.100
  ```

  Ejemplo de Salida (fragmento):

  ```
  PORT   STATE SERVICE
  22/tcp open  ssh
  80/tcp open  http
  ```

- Escaneo de puertos específicos:

  ```
  nmap -p 22,80,443 192.168.1.100
  ```

- Detección de versiones de servicios (más verboso):

  ```
  nmap -sV 192.168.1.100
  ```

  Ejemplo de Salida (fragmento):

  ```
  PORT   STATE SERVICE VERSION
  22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.7 (Ubuntu Linux; protocol
  2.0)
  80/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
  ```

- Detección del sistema operativo:

  ```
  nmap -O 192.168.1.100
  ```

  Nota: La detección de SO no siempre es 100% precisa, pero da una buena aproximación.

Conceptos Clave para la Resolución de Problemas

- Modelo OSI/TCP/IP: Una buena comprensión de estas capas te permite aislar problemas. Por ejemplo, si `ping` funciona (Capa 3 - IP) pero no puedes acceder a un sitio web (Capa 7 - Aplicación), el problema no está en la conectividad básica sino probablemente en el firewall, el servidor web o DNS.

- Diagnóstico Paso a Paso (Bottom-Up Approach):

1. Capa Física/Enlace (Capa 1/2): ¿Está la interfaz activa y conectada? (`ip link show`, revisar cables).
2. Capa de Red (Capa 3 - IP): ¿Tienes la dirección IP correcta? ¿La máscara de red es correcta? ¿Hay una ruta al destino? (`ip a`, `ip r`, `ping`, `traceroute`).
3. Capa de Transporte (Capa 4 - TCP/UDP): ¿El puerto está abierto en el destino? ¿Hay un firewall intermedio bloqueando el tráfico? (`ss`, `netstat`, `nc`, `nmap`).
4. Capa de Aplicación (Capa 7): ¿El servicio (web server, SSH daemon) está corriendo y configurado correctamente? (`lsof`, `systemctl status <service>`).

- Firewalls: Los firewalls (como iptables/nftables, firewalld en Red Hat/Rocky, o ufw en Ubuntu) son una causa muy común de problemas de conectividad. Siempre verifica las reglas del firewall si un servicio no es accesible. Un `tcpdump` puede mostrar si el tráfico llega al sistema pero no es respondido, lo que a menudo apunta a un firewall.