# 📁 LPIC-2 / 🔒 Examen 212 - Seguridad del Sistema

Este examen cubre la configuración de seguridad, incluyendo enrutamiento, firewalls, seguridad de servicios y detección de intrusiones.

### 212.1 Configurar un router

**Teoría**

Un router es un dispositivo de red que reenvía paquetes de datos entre diferentes redes IP. Un sistema Linux con múltiples interfaces de red conectadas a diferentes segmentos de red puede funcionar como un router habilitando la funcionalidad de reenvío de paquetes en el kernel.

**Componentes Clave para Enrutamiento en Linux:**

1. **Reenvío de IP en el Kernel (IP Forwarding):**
    - Esta es la función básica que permite al kernel recibir paquetes en una interfaz y enviarlos por otra si el destino no es local.
    - **Verificar Estado:**
        - `cat /proc/sys/net/ipv4/ip_forward` (muestra `0` para deshabilitado, `1` para habilitado).
        - `sysctl net.ipv4.ip_forward` (muestra el mismo valor).
    - **Habilitar Temporalmente:** `sudo echo 1 > /proc/sys/net/ipv4/ip_forward`. Se pierde al reiniciar.
    - **Habilitar Persistentemente:** Añadir o asegurar que la línea `net.ipv4.ip_forward = 1` (o `net.ipv6.conf.all.forwarding = 1` para IPv6) está presente en un archivo de configuración de sysctl (ej: `/etc/sysctl.conf` o un archivo `.conf` en `/etc/sysctl.d/` - Revisado en 206.3). Aplicar cambios persistentes con `sudo sysctl -p`.
2. **Tabla de Enrutamiento:**
    - El kernel mantiene una tabla de enrutamiento que contiene reglas para decidir el "siguiente salto" (next hop) de un paquete basándose en su dirección IP de destino.
    - **Ver Tabla:** `ip route show` o `route -n` (ambos muestran la tabla principal).
    - **Ejemplo de Entradas:**
        - Rutas a redes locales conectadas directamente (interfaz).
        - Ruta por defecto (`default via <gateway> dev <interfaz>`) para destinos desconocidos (generalmente apuntando al router principal de la red).
        - Rutas estáticas añadidas manualmente o por demonios de enrutamiento.
3. **Enrutamiento Estático:**
    - Añadir rutas específicas manualmente a la tabla de enrutamiento.
    - **Añadir Ruta:** `sudo ip route add <red_destino>/<mascara> via <ip_siguiente_salto> [dev <interfaz>]` (Revisado en 205.3).

- **Eliminar Ruta:** `sudo ip route del <red_destino>/<mascara>` (Revisado en 205.3).
- Las rutas estáticas añadidas manualmente se pierden al reiniciar a menos que se hagan persistentes a través de archivos de configuración de red o scripts de inicio.

4. **Enrutamiento Dinámico:**
    - Los routers intercambian información de enrutamiento con otros routers utilizando protocolos de enrutamiento dinámico (RIP, OSPF, BGP) para aprender automáticamente sobre las redes y construir/actualizar sus tablas de enrutamiento.
    - **Demonios de Enrutamiento:** Se necesita software en el espacio de usuario para implementar estos protocolos.
        - **Quagga:** Un demonio de enrutamiento popular y de código abierto que soporta varios protocolos.
        - **FRR (Free Range Routing):** El sucesor moderno y activo de Quagga, más potente.
    - **Paquetes (Diferencias):** `quagga` o `frr`.
    - **Configuración:** Se configuran a través de archivos específicos (ej: `zebra.conf`, `ripd.conf`, `ospfd.conf` en `/etc/quagga/` o `/etc/frr/`) utilizando una interfaz de línea de comandos similar a los routers comerciales (VTY - Virtual Terminal Interface). **La configuración de protocolos dinámicos es avanzada.**
5. **Network Address Translation (NAT):**
    - Permite a dispositivos en una red privada (con IPs privadas) acceder a Internet (con IPs públicas) compartiendo una o más IPs públicas.
    - **Tipos Comunes:**
        - **SNAT (Source NAT):** Modifica la dirección IP *origen* del paquete. Comúnmente usado como **Masquerading**, que es una forma de SNAT donde la IP origen saliente es la IP de la interfaz de salida del router, útil para IPs dinámicas. Permite a múltiples clientes internos compartir una IP pública para salir a Internet.
        - **DNAT (Destination NAT):** Modifica la dirección IP *destino* del paquete. Comúnmente usado para publicar servicios internos a Internet (ej: redirigir tráfico entrante al puerto 80 de la IP pública del router al puerto 80 de un servidor web en la red privada).
    - **Implementación:** NAT se implementa usando las reglas de **firewall** (principalmente `iptables` o `firewalld`). Se usa la tabla `nat`.
        - **Regla para MASQUERADE (con `iptables`):** `sudo iptables -t nat -A POSTROUTING -o <interfaz_salida_publica> -j MASQUERADE`. Esto reescribe la IP origen de los paquetes que salen por la interfaz pública para que parezca que vienen del router.

- **Configuración con `firewalld`:** Se habilita el masquerading en la zona de la interfaz de salida a Internet (ej: `sudo firewall-cmd --zone=<zona_publica> --add-masquerade --permanent`).
- **Configuración con `ufw`:** Requiere editar archivos de reglas personalizados.

6. **Firewall:**

   - Un router Linux generalmente actúa también como firewall para controlar qué tráfico se permite o deniega entre las redes conectadas.
   - Se configuran reglas en las cadenas `INPUT` (tráfico al propio router), `OUTPUT` (tráfico originado por el router) y **`FORWARD`** (tráfico que PASA A TRAVÉS del router entre interfaces). La cadena `FORWARD` es clave para controlar el tráfico enrutado.
   - Herramientas: `iptables`, `firewalld`, `ufw` (Revisado en 212.2).

**Consideraciones para Configurar un Router Linux:**

- El sistema debe tener al menos dos interfaces de red, cada una configurada en una subred IP diferente.
- El reenvío de IP debe estar habilitado.
- Las tablas de enrutamiento deben tener las rutas correctas para llegar a las redes de destino.
- El firewall debe permitir el tráfico necesario y aplicar NAT si es requerido.
- Si se usa NAT, los clientes en la red privada deben usar la IP de la interfaz interna del router Linux como su puerta de enlace predeterminada.