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

212.3 Asegurar servicios

Teoría

Cada servicio de red que se ejecuta en un sistema Linux (servidor web, servidor SSH, servidor DNS, etc.) representa una superficie de ataque potencial. Asegurar estos servicios es tan importante como tener un firewall robusto.

Principios Generales para Asegurar Servicios:

1. **Minimización:**
    - **Ejecutar solo lo necesario:** Deshabilita o desinstala cualquier servicio de red que no sea estrictamente necesario. Cuantos menos servicios corran, menor será la superficie de ataque.
    - **Verificar servicios activos:** Usa comandos como `ss -tulnp` o `netstat -tulnp` para ver qué puertos están abiertos y qué programas están escuchando.
2. **Ejecución con Privilegios Mínimos:**
    - Los servicios (demonios) nunca deben ejecutarse como el usuario `root` a menos que sea absolutamente indispensable (generalmente, solo para iniciar y vincular a puertos privilegiados <1024, y luego cambian de usuario).
    - La mayoría de los servicios se configuran para ejecutarse como un usuario dedicado no privilegiado (ej: `www-data` para Apache/Nginx en Debian, `apache` para Apache en Red Hat, `bind` para BIND, `postfix` para Postfix). Esto limita lo que un atacante puede hacer si compromete el servicio.
3. **Endurecimiento (Hardening) de la Configuración:**
    - **Deshabilitar Funciones Inseguras:** Desactiva características obsoletas o inseguras en la configuración del servicio (ej: deshabilitar Telnet, rlogin; deshabilitar versiones antiguas de SSL/TLS como TLSv1.0, ciphers débiles; deshabilitar acceso anónimo en Samba/NFS; deshabilitar autenticación por contraseña para root en SSH).
    - **Control de Acceso Basado en Origen:** Configura el servicio para aceptar conexiones solo de direcciones IP o redes de confianza (además del firewall). Esto se hace en la configuración específica del servicio (ej: directivas `Allow`/`Deny` en Apache, `listen` en Nginx/Postfix, `allow`/`reject` en `/etc/hosts.allow`/`/etc/hosts.deny` si el servicio usa libwrap, ACLs en Samba/NFS/LDAP/DHCP).
    - **Autenticación Fuerte:** Si el servicio requiere autenticación, configura métodos fuertes (ej: autenticación por clave pública en SSH, no permitir contraseñas triviales, usar PAM para políticas de contraseña - Revisado 210.2).

    - **Registro Detallado:** Configura el servicio para registrar eventos relevantes (conexiones, autenticaciones fallidas, errores) con un nivel de detalle apropiado para la monitorización y auditoría.
4. **Actualizaciones de Software:**
    - Mantén el software del servicio y todas sus dependencias (librerías) actualizadas. Las actualizaciones a menudo incluyen parches de seguridad para vulnerabilidades descubiertas.
5. **Cifrado:**
    - Utiliza cifrado (TLS/SSL) para cualquier comunicación que implique datos sensibles (credenciales, información privada). Configura servicios como HTTPS, LDAPS, SMTPS, SPOP3, SIMAP, SSH (aunque SSH cifra por defecto, la configuración de algoritmos es relevante). (Revisado 208.4 para HTTPS, 210.3/210.4 para LDAPS, 211.2 para SMTPS).
6. **Límites de Recursos:**
    - Configura límites de recursos para los servicios para prevenir ataques de denegación de servicio (DoS) que busquen agotar CPU, memoria o conexiones. Esto se puede hacer a nivel de sistema (`pam_limits.so` - Revisado 210.2) o en la configuración específica del servicio (ej: `MaxClients` en Apache, `worker_connections` en Nginx).
7. **Monitorización y Auditoría:**
    - Revisa regularmente los archivos de log de los servicios y los logs de autenticación del sistema (`auth.log`/`secure`) en busca de actividad sospechosa (intentos de login fallidos, escaneos de puertos, errores inusuales).
8. **Mecanismos de Seguridad a Nivel de Sistema (Control de Acceso Obligatorio - MAC):**
    - **SELinux (Security-Enhanced Linux):** (Común en Red Hat/CentOS/Fedora) Un framework MAC que restringe lo que los procesos pueden hacer basándose en un contexto de seguridad definido por políticas, *independientemente* de los permisos de usuario/grupo tradicionales (DAC - Discretionary Access Control). Los servicios se ejecutan en dominios SELinux específicos con permisos muy limitados (ej: un servidor web solo puede leer de `/var/www/html`). Configurar SELinux para servicios personalizados es complejo (`semanage`, `restorecon`, `chcon`).
    - **AppArmor:** (Común en Debian/Ubuntu/SUSE) Otro framework MAC basado en perfiles. Define qué recursos del sistema (archivos, capacidades de red) puede o no puede acceder un programa específico. Los servicios se ejecutan bajo perfiles de AppArmor. Es generalmente más simple de configurar que SELinux. (`aa-status`, `aa-enforce`, `aa-complain`).
    - Ambos mecanismos añaden una capa de defensa profunda crucial.

Asegurar Servicios Comunes (Ejemplos):

- **SSH (`sshd`):**
  - Archivo de configuración: `/etc/ssh/sshd_config`.
  - Directivas clave: `Port` (cambiar puerto por defecto), `PermitRootLogin no`, `PasswordAuthentication no` (si usas autenticación por clave), `AllowUsers` / `AllowGroups` (limitar quién puede loguearse), `PubkeyAuthentication yes`, `Protocol 2` (deshabilitar SSHv1).
- **Servidores Web (Apache/Nginx):**
  - Configuración HTTPS (Revisado 208.4).
  - Permisos de archivos y directorios en el Document Root.
  - Deshabilitar módulos o directivas innecesarias.
  - Restringir acceso a directorios sensibles o URLs.
  - Limitar tamaño de peticiones, conexiones.
- **Servidor DNS (BIND - `named`):**
  - Restringir consultas y recursión (Revisado 207.3).
  - Restringir transferencias de zona.
  - Ejecutar en chroot y con usuario dedicado.
- **Samba (`smbd`, `nmbd`):**
  - Deshabilitar acceso `guest ok`.
  - Usar `valid users` para restringir acceso a comparticiones (Revisado 209.1).
  - Configurar permisos de archivo/directorio en el servidor Linux subyacente.
- **NFS (`nfsd`):**
  - Usar `root_squash` (default y recomendado) o `all_squash` en `/etc/exports`. Evitar `no_root_squash`.
  - Restringir clientes por IP/red en `/etc/exports` (Revisado 209.3).