📁 LPIC-2 / 🌍 Examen 208 - Servicios Web - Ejercicios

*Nota: Estos ejercicios implican generar claves/certificados y modificar configuraciones sensibles. Realízalos **SIEMPRE en una VM de prueba dedicada**. Asegúrate de que tu firewall permite tráfico en el puerto 443. Necesitarás privilegios de superusuario (`sudo`). Usaremos un certificado autofirmado para la práctica.*

Ejercicio 8.4.1: Verificando Reglas de Firewall para el Puerto 443

- **Objetivo:** Asegurarse de que el firewall permite el tráfico web seguro (HTTPS).
- **Requisitos:** Privilegios de superusuario (`sudo`). Identificar la herramienta de firewall activa (Ej. 5.2.5).
- **Desarrollo Paso a Paso:**
    1. Abre una terminal.
    2. **Si usas `firewalld`:** Ejecuta `sudo firewall-cmd --zone=<zona> --list-services` o `sudo firewall-cmd --zone=<zona> --list-ports`. Busca el servicio `https` o el puerto `443/tcp`. Si no está, añádelo: `sudo firewall-cmd --zone=<zona> --add-service=https --permanent` y `sudo firewall-cmd --reload`.
    3. **Si usas `ufw`:** Ejecuta `sudo ufw status`. Busca reglas para el puerto 443 TCP. Si no están, añádelas: `sudo ufw allow 443/tcp`.
    4. **Si usas `iptables` directamente:** Ejecuta `sudo iptables -L -v -n`. Busca reglas que permitan tráfico entrante a puerto 443 TCP en la cadena `INPUT`.

Ejercicio 8.4.2: (Conceptual) Generando una Clave Privada y un Certificado Autofirmado

- **Objetivo:** Entender cómo crear archivos de certificado para pruebas.
- *Requisitos:* Acceso a la línea de comandos. El paquete `openssl` instalado (suele estar por defecto). Privilegios de superusuario (`sudo`) para guardar en `/etc/ssl/`. **VM de prueba.** Saber el nombre de host que usarás para acceder a la VM (ej: la IP o un nombre en `/etc/hosts`).
- **Desarrollo Paso a Paso:**
    1. Abre una terminal.
    2. **Genera la clave privada y el certificado autofirmado en un paso:** Ejecuta `sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/mywebserver.key -out /etc/ssl/certs/mywebserver.crt`.
        - `-x509`: Crea un certificado autofirmado.
        - `-nodes`: No proteger la clave privada con contraseña (más fácil para servidores web).
        - `-days 365`: El certificado será válido por 365 días.
        - `-newkey rsa:2048`: Crea una nueva clave privada RSA de 2048 bits.
        - `-keyout ...`: Archivo de salida para la clave privada.

    - `-out ...`: Archivo de salida para el certificado.
3. **Te pedirá información para el certificado (Common Name):** El más importante es "Common Name (e.g. server FQDN or YOUR name)". **Introduce aquí la IP o el nombre de host exacto que usarás para acceder al servidor (ej: la IP de tu VM).** Si no coincide, el navegador mostrará un error de nombre inválido.
4. **Verifica que los archivos fueron creados:** Ejecuta `sudo ls -l /etc/ssl/private/mywebserver.key` y `sudo ls -l /etc/ssl/certs/mywebserver.crt`. La clave privada debe tener permisos restrictivos (solo lectura para root, sin acceso para otros).

Ejercicio 8.4.3: (Conceptual) Configurando Apache para HTTPS con Certificado Autofirmado

- **Objetivo:** Habilitar SSL/TLS en Apache usando los archivos generados.
- *Requisitos:* Apache instalado y corriendo. Certificado autofirmado generado. Privilegios de superusuario (`sudo`). **VM de prueba.**
- **Desarrollo Paso a Paso (Conceptual):**
    - **Apache (Debian/Ubuntu):**
        1. Habilita el módulo SSL: `sudo a2enmod ssl`.
        2. Habilita la configuración SSL por defecto (contiene un Virtual Host de ejemplo para 443): `sudo a2enconf ssl`.
        3. Edita el archivo de configuración del Virtual Host SSL por defecto o crea uno nuevo (`sudo vi /etc/apache2/sites-available/default-ssl.conf` o similar).
        4. Busca el bloque `<VirtualHost *:443>`.
        5. Asegúrate de que `SSLEngine On` está presente.
        6. Actualiza las rutas de los archivos de clave y certificado para que apunten a tus archivos generados:
           Apache
           ```
           SSLCertificateFile      /etc/ssl/certs/mywebserver.crt
           SSLCertificateKeyFile   /etc/ssl/private/mywebserver.key
           # No necesitamos SSLCACertificateFile para un certificado autofirmado
           ```
        7. Guarda y sal.
        8. Verifica la sintaxis de Apache: `sudo apache2ctl configtest`.
        9. Recarga Apache: `sudo systemctl reload apache2`.
    - **Apache (Red Hat/CentOS/Fedora):**
        1. Instala el módulo SSL si es necesario: `sudo dnf install mod_ssl`. Esto suele crear un archivo `/etc/httpd/conf.d/ssl.conf`.
        2. Edita el archivo `ssl.conf` (`sudo vi /etc/httpd/conf.d/ssl.conf`).
        3. Busca el bloque `<VirtualHost _default_:443>`.

4. Asegúrate de que `SSLEngine On` está presente.
5. Actualiza las rutas de los archivos de clave y certificado:
   Apache

```
SSLCertificateFile /etc/ssl/certs/mywebserver.crt
SSLCertificateKeyFile /etc/ssl/private/mywebserver.key
# SSLCACertificateFile no es necesario
```

6. Guarda y sal.
7. Verifica la sintaxis de Apache: `sudo httpd -t`.
8. Recarga Apache: `sudo systemctl reload httpd`.

Ejercicio 8.4.4: (Conceptual) Configurando Nginx para HTTPS con Certificado Autofirmado

- **Objetivo:** Habilitar SSL/TLS en Nginx usando los archivos generados.
- *Requisitos:* Nginx instalado y corriendo. Certificado autofirmado generado. Privilegios de superusuario (`sudo`). **VM de prueba.**
- **Desarrollo Paso a Paso (Conceptual):**
  1. Abre una terminal.
  2. Edita el archivo de configuración del Server Block por defecto o crea uno nuevo (ej: `sudo vi /etc/nginx/sites-available/default` o un archivo en `conf.d/`).
  3. Busca el bloque `server` para el puerto 80 y (opcionalmente) añade una redirección a HTTPS. Luego, crea o modifica un bloque `server` para el puerto 443.
     Nginx

```
# Bloque para redirigir HTTP a HTTPS (opcional)
server {
    listen 80;
    server_name <nombre_host_o_ip>;
    return 301 https://$host$request_uri;
}

# Bloque para HTTPS
server {
    listen 443 ssl; # Escucha en 443 y habilita SSL
    server_name <nombre_host_o_ip>;

    ssl_certificate /etc/ssl/certs/mywebserver.crt; # Ruta al
certificado
    ssl_certificate_key /etc/ssl/private/mywebserver.key; # Ruta a
la clave privada

    # Opcional: Configuraciones SSL recomendadas (ej: protocolos,
ciphers)
    # ssl_protocols TLSv1.2 TLSv1.3;
    # ssl_ciphers 'EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH';
    # ssl_prefer_server_ciphers on;

    root /var/www/html; # O el Document Root para este sitio
    index index.html index.htm;
```

```
    # Logs (opcional, si quieres logs separados para HTTPS)
    # error_log /var/log/nginx/ssl_error.log;
    # access_log /var/log/nginx/ssl_access.log;
}
```

- Reemplaza `<nombre_host_o_ip>` con la IP o nombre de host que usarás para acceder.

4. Guarda y sal.
5. Verifica la sintaxis de Nginx: `sudo nginx -t`.
6. Recarga Nginx: `sudo systemctl reload nginx`.

Ejercicio 8.4.5: Probando Acceso HTTPS con Certificado Autofirmado

- **Objetivo:** Verificar que el servidor web está respondiendo en HTTPS, aceptando la advertencia de seguridad.
- **Requisitos:** Configuración HTTPS aplicada y recargada/reiniciada. Puerto 443 abierto en firewall. Nombre común del certificado coincide con la IP/nombre de host usado para acceder.
- **Desarrollo Paso a Paso:**
  1. Abre un navegador y accede a `https://<ip_de_tu_vm>` o `https://<nombre_host_de_tu_vm>`.
  2. **Deberías ver una advertencia de seguridad** indicando que el certificado no es de confianza. Procede a aceptar la advertencia (los pasos exactos dependen del navegador).
  3. Deberías ver la página web por defecto.
  4. **Usa `curl` ignorando la verificación del certificado:** Ejecuta `curl -vk https://<ip_de_tu_vm>`. La opción `-k` deshabilita la verificación del certificado, permitiéndote probar la conexión TLS/SSL. `-v` muestra detalles del handshake.

Ejercicio 8.4.6: (Conceptual) Proceso con Certificado de CA de Confianza

- **Objetivo:** Entender conceptualmente cómo sería el proceso con un certificado real.
- **Requisitos:** Acceso a la línea de comandos. `openssl` instalado.
- **Desarrollo Paso a Paso (Conceptual):**
  1. **Generar Clave Privada:** `openssl genrsa -out mydomain.key 2048`.
  2. **Crear CSR:** `openssl req -new -key mydomain.key -out mydomain.csr`. Rellena la información, asegurándote de que el "Common Name" es el FQDN exacto de tu sitio (ej: `www.example.com`).
  3. **Enviar CSR a CA y Recibir Certificado/Cadena:** (Este paso ocurre fuera de la línea de comandos, interactuando con el sitio web de la CA). Recibes `mydomain.crt` (tu certificado público) y potencialmente `chain.pem` (certificados intermedios).
  4. **Configurar Servidor Web:** Modifica `SSLCertificateFile`/`ssl_certificate` para apuntar a `mydomain.crt` y

`SSLCertificateKeyFile`/`ssl_certificate_key` para apuntar a `mydomain.key`. Configura `SSLCACertificateFile` (Apache) o asegúrate de que `mydomain.crt` contiene la cadena completa (Nginx a veces requiere concatenar `cat mydomain.crt chain.pem > mydomain-bundle.crt` y usar este archivo).

5. **Recargar/Reiniciar:** Aplica los cambios.
6. **Prueba:** Accede en un navegador. No debería haber advertencias si la cadena es correcta y la CA es de confianza.