Trabajo Final
Laboratorio de Redes y Sistemas Operativos

UNQ

2025

Participantes:

Melina Iglesias

Valentina Gallippi

Marcos Roldan

1. Descripción del Proyecto

El proyecto consistió en la implementación de un servidor de medios personal utilizando la plataforma Jellyfin. Jellyfin es una aplicación de código abierto que permite a los usuarios organizar, administrar y transmitir (streamear) su colección de películas, series y otros contenidos multimedia. El objetivo principal fue montar una solución de "Netflix" propio accesible desde la red local.

El proyecto abordó los siguientes aspectos clave:

- Servicio de Streaming: Configuración de un servidor de medios para permitir el acceso a la colección de forma remota (dentro de la red local).
- Redes: Gestión del acceso a través de puertos específicos, lo cual es fundamental para el streaming de video y la conectividad de múltiples clientes (celulares, Smart TV, otras PCs).
- Seguridad: Implementación de Fail2ban para proteger el servidor de ataques de fuerza bruta en los intentos de autenticación.
- Acceso Web: Uso de Caddy como proxy inverso para facilitar el acceso web al servicio.

2. Relevamiento de Características de la Máquina Utilizada

Tipo de Máquina: Máquina Virtual (VM).
CPU Asignada: 4 procesadores (núcleos virtuales).
RAM Asignada: 8 GB.

3. Relevamiento de Sistema Operativo y Software Utilizado y/o Necesario

Sistema Operativo: **Ubuntu** Plataforma base sobre la cual se ejecutaron todos los servicios.

Servidor Multimedia: **Jellyfin** Software encargado de la gestión y transcodificación de medios.

Proxy Inverso: **Caddy** es un servidor web moderno y extensible de código abierto, escrito en el lenguaje Go, conocido por su seguridad por defecto y su facilidad de configuración. En el marco de este proyecto, Caddy se implementó cumpliendo la función de **Proxy Inverso**. Actúa como un intermediario situado entre los dispositivos de los usuarios (clientes) y el servidor Jellyfin. Su tarea consiste en interceptar las solicitudes entrantes en un puerto designado (5001) y redirigirlas internamente hacia el servicio de medios (puerto 8096), gestionando el tráfico de red de manera eficiente y ocultando la infraestructura interna del servicio.

Seguridad: **Fail2ban** Fail2ban es un framework de prevención de intrusiones diseñado para sistemas Linux. Su función principal es proteger el servidor contra ataques de fuerza bruta y accesos no autorizados mediante el monitoreo continuo de los archivos de registro (logs) de los servicios activos. En el contexto de este proyecto, Fail2ban opera realizando las siguientes acciones:

1. **Monitoreo de registros:** Analiza en tiempo real los logs del servicio buscando patrones de intentos fallidos.
2. **Identificación de ataques:** Utiliza expresiones regulares para identificar la IP de origen de un ataque.
3. **Bloqueo de IP:** Interactúa con el firewall (iptables) para bloquear la dirección IP maliciosa.
4. **Desbloqueo automático:** Tras un tiempo configurado, libera la IP a menos que la actividad sospechosa persista.

4. Instructivo Paso a Paso de lo Ocurrido (Ejecución)

A continuación, se detalla el proceso de instalación y configuración del servidor:

4.1. Instalación de Jellyfin (Servidor de Medios)

1. Se instalaron las dependencias necesarias: curl y gnupg (sudo apt install curl gnupg).
2. Se habilitó el repositorio Universe para obtener las dependencias de FFmpeg (sudo add-apt-repository universe).
3. Se descargó e instaló la clave de firma GPG del equipo Jellyfin.
    - sudo mkdir -p /etc/apt/keyrings
    - curl -fsSL ... | sudo gpg --dearmor -o /etc/apt/keyrings/jellyfin.gpg
4. Se agregó la configuración del repositorio de Jellyfin a la lista de fuentes APT.
5. Se actualizaron los repositorios APT (sudo apt update).
6. Se instaló el metapaquete de Jellyfin (sudo apt install jellyfin).
7. Se verificó el acceso inicial a través del puerto por defecto: http://localhost:8096.

4.2. Instalación y Configuración de Caddy (Proxy Inverso)

1. Se instaló Caddy siguiendo la secuencia de comandos para agregar su repositorio e instalar el paquete.
2. Se configuró el proxy inverso utilizando el comando: caddy reverse-proxy --from :5001 --to 127.0.0.1:8096
3. Se verificó el acceso a través del puerto del proxy: http://localhost:5001.

4.3. Instalación y Configuración de Fail2ban (Seguridad)

1. Se instaló Fail2ban (sudo apt install fail2ban).
2. Se creó el archivo de cárcel para Jellyfin: /etc/fail2ban/jail.d/jellyfin.local.
    - Se definieron los puertos (port = 8096,5001), el número máximo de reintentos (maxretry = 3), el tiempo de baneo (bantime = 86400) y se

especificó la ruta de los logs: /var/log/jellyfin/jellyfin(númerodearchivo).log.

3. Se creó el archivo de filtro para Jellyfin: /etc/fail2ban/filter.d/jellyfin.conf.
    - Se pegó la expresión regular (failregex) para identificar intentos de autenticación fallidos: ^.*Authentication request for .* has been denied \(IP: "<ADDR>"\)\.
4. Se reinició Fail2ban para aplicar la configuración (sudo systemctl restart fail2ban).
5. Se verificó el estado de Fail2ban para confirmar su ejecución (sudo systemctl status fail2ban).

4.4. Configuraciones Adicionales

1. Red: Se configuró la Máquina Virtual en Modo Bridge para que sea accesible por su IP en la red local.
2. Archivos Multimedia: Se otorgaron los permisos necesarios al usuario de Jellyfin para que pueda acceder al directorio donde se encuentran las películas/series a streamear.

5. Problemas que Surgieron y sus Soluciones

Durante la implementación del servidor Jellyfin, encontramos tres desafíos principales que requerían soluciones específicas para el correcto funcionamiento del sistema.

Problema 1: Fallo en la Instalación Manual desde Fuentes

Se intentó compilar el servidor manualmente desde el código fuente como se solicitó. Sin embargo, la documentación del repositorio estaba desactualizada respecto a la versión del código actual, lo que generó errores insalvables durante el proceso de construcción (*build*). **Solución:** Debido a la inconsistencia en el repositorio fuente, se optó por realizar la instalación estándar utilizando el gestor de paquetes de Ubuntu (`apt`), lo que permitió desplegar el servicio de forma correcta y estable.

Problema 2: Acceso Inaccesible desde la Red Local

El primer inconveniente fue que el servidor Jellyfin, aunque funcionaba correctamente dentro de la VM (accesible vía localhost), no era visible ni accesible desde otros dispositivos de la red local (celulares o PCs anfitrionas).

Solución: Se identificó que la Máquina Virtual no estaba configurada para interactuar directamente con la red física. La solución fue cambiar el modo de red de la VM al Modo Bridge (Puente). Esta configuración le permitió a la VM obtener una dirección IP propia dentro de la red local, haciéndola visible y accesible para el *streaming* desde cualquier cliente conectado a la misma red.

Problema 3: El Servidor Jellyfin No Podía Acceder a los Archivos Multimedia

Tras la instalación y configuración de la interfaz web de Jellyfin, la biblioteca permanecía vacía, a pesar de que los archivos de video estaban físicamente en el disco de la VM.

Solución: Este fue un conflicto clásico de permisos de usuario en Linux. El servicio Jellyfin se ejecuta bajo un usuario específico que carecía de los permisos de lectura y ejecución necesarios para acceder al directorio donde se almacenaban las películas y series. Se resolvió ajustando recursivamente los permisos del directorio multimedia, asegurando que el usuario del servicio Jellyfin tuviera los permisos adecuados para leer y listar el contenido (típicamente utilizando comandos como chown o chmod).

Problema 4: Configuración Incompleta de Fail2ban

Al configurar Fail2ban para proteger los puertos de Jellyfin (8096 y 5001), el servicio no lograba banear IPs maliciosas, a pesar de haber intentado fallar la autenticación.

Solución: La causa fue un error en la definición de la ruta del archivo de *logs* (logpath) en la cárcel (jail.d/jellyfin.local) de Fail2ban. El nombre del archivo de log de Jellyfin usualmente incluye un número variable. Se verificó la ruta exacta del log

en /var/log/jellyfin/ y se corrigió el parámetro logpath para que apuntara correctamente al archivo de registro (ej. jellyfin(númerodearchivo).log), permitiendo a Fail2ban monitorear y aplicar el filtro de *failregex* de manera efectiva.

6. Uso de Inteligencia Artificial (IA)

La Inteligencia Artificial (IA) se utilizó exclusivamente como herramienta de soporte para la resolución de conflictos y la verificación de sintaxis durante la etapa de depuración del proyecto, tal como se mencionó en la descripción inicial.

El modelo utilizado fue ChatGPT.

El uso principal se centró en dos áreas críticas:

1. Resolución de Conflictos de Permisos: Cuando Jellyfin no reconocía los archivos, la IA fue consultada con *prompts* como: *"Jellyfin no carga la biblioteca aunque los archivos están ahí, ¿qué permisos necesito configurar?"*. Esto permitió obtener la sintaxis precisa para los comandos chown y chmod requeridos para otorgar permisos al usuario de servicio de Jellyfin.
2. Sintaxis de Configuración (Fail2ban y Caddy): Se utiliza para verificar y generar la sintaxis correcta en archivos de configuración sensibles. Por ejemplo, al implementar Fail2ban, se consultó con el *prompt*: *"El filtro de Fail2ban para Jellyfin no funciona, necesito un failregex para 'Authentication request for … denied (IP: "…")."*. De igual manera, se consultó para confirmar la sintaxis más sencilla del *reverse proxy* de Caddy.

En resumen, la IA fue una herramienta de consulta técnica rápida que aceleró la depuración, asegurando la precisión en comandos y expresiones regulares que son propensas a errores manuales.