

Gestión y Visualización en Directo de Cámaras de Seguridad



- Autor: Victor Barroso
- Tutor: Marc Crespo
- Curso: 2ºASIR
- Fecha: 2023/2024



Resumen

Resumen: El informe describe el proceso de desarrollo de una infraestructura de visualización de cámaras IP. El objetivo de esta solución es ofrecer tanto a empresas como a particulares una herramienta para monitorizar eficazmente su entorno. Esto ayuda a abordar cualquier problema que pueda surgir al intentar configurar dicha red, incluidas consideraciones sobre la escalabilidad para satisfacer sus necesidades específicas, ya sean ascendentes o descendentes. El objetivo final de esta plataforma es garantizar la facilidad de escalabilidad y control y, al mismo tiempo, mantener la seguridad para los entornos de todos los usuarios. Esto debería poder adaptarse fácilmente a diversos requisitos de diferentes usuarios después de la implementación con capacidades, como control y seguridad efectivos, garantizados independientemente de las demandas específicas que se requieran durante ese período.

Resum: L'informe descriu el procés de desenvolupament d'una infraestructura de visualització de càmeres IP. L'objectiu d'esta solució és oferir tant a empreses com a particulars una ferramenta per a monitorar eficaçment el seu entorn. Això ajuda a abordar qualsevol problema que puga sorgir en intentar configurar esta xarxa, incloses consideracions sobre l'escalabilitat per a satisfer les seues necessitats específiques, ja siguen ascendents o descendents. L'objectiu final d'esta plataforma és garantir la facilitat d'escalabilitat i control i, al mateix temps, mantindre la seguretat per als entorns de tots els usuaris. Això hauria de poder adaptar-se fàcilment a diversos requisits de diferents usuaris després de la implementació amb capacitats, com a control i seguretat efectius, garantits independentment de les demandes específiques que es requerisquen durant eixe període.

Abstract: The report describes the development process of an IP camera viewing infrastructure. The objective of this solution is to offer both companies and individuals a tool to effectively monitor their environment. This helps address any issues that may arise when trying to configure such a network, including considerations about scalability to meet your specific needs, whether upstream or downstream. The ultimate goal of this platform is to ensure ease of scalability and control while maintaining security for all users' environments. This should be easily adaptable to various requirements of different users after implementation with capabilities, such as effective control and security, guaranteed regardless of the specific demands required during that period.

Índice

1. Introducción.....	5
1.1. Módulos a los que implica.....	6
1.2. Breve descripción del proyecto.....	6
2. Diseño de red.....	7
3. Base de datos.....	8
4. WEB.....	9
5. Configuración de cámaras.....	13
5.1. Integración de cámaras en la web.....	17
6. Configuración Servidor 1.....	19
6.1. Tarjetas de red.....	19
6.2. Configuración Apache2.....	19
6.3. Docker.....	21
6.3.1. Dockerfile.....	21
6.3.2. Docker compose.....	24
6.4. Servidor de variables de sesion php.....	25
6.5. Balanceador de carga.....	26
7. Configuración servidor 2.....	28
7.1. tarjetas de red.....	28
7.2. Configuración DNS.....	29
8. Mikrotik.....	30
8.1. ¿Qué es un router Mikrotik?.....	30
8.2. Instalación y configuración inicial.....	30
8.3. Montar VPN.....	32
8.4. Conexión a la VPN.....	40
9. Recursos.....	43
9.1. Herramientas hardware.....	43
9.2. Herramientas software.....	43
10. Conclusiones (Cambiar).....	44
10.1. Conclusiones sobre el trabajo realizado.....	44
10.2. Conclusiones personales.....	44
10.3. Posibles ampliaciones y mejoras.....	44
11. Anexos y documentos complementarios.....	45
12. Webgrafía.....	45

1. *Introducción*

En el entorno contemporáneo, la seguridad y la vigilancia se han convertido en preocupaciones importantes tanto a nivel individual como corporativo. En este contexto, las cámaras IP surgieron como una solución tecnológica versátil y avanzada para el seguimiento y vigilancia remotos.

¿Qué es una cámara IP?

La cámara IP es la abreviatura de “Cámara de protocolo de Internet” y es un dispositivo de vigilancia conectado a una red. Esto le otorga una serie de ventajas importantes, una de ellas es la posibilidad de acceder en remoto desde un ordenador con Internet. Estas cámaras suelen proporcionar imágenes y videos de gran calidad, además de características adicionales como la capacidad de detectar movimiento, visión nocturna y audio de doble vía dependiendo de la cámara.

¿Qué protocolos suelen usar?

- **ONVIF** (Foro de interfaz de video en red abierta): este estándar de la industria está diseñado para mejorar la interoperabilidad entre varios dispositivos de seguridad, como cámaras IP, grabadoras de video en red (NVR) y software de administración de video (VMS).
- **RTSP** (Protocolo de transmisión en tiempo real): se utiliza para la transmisión en tiempo real de audio y video a través de redes IP. Le permite crear y monitorear flujos de datos sincronizados, ideal para disfrutar de video en vivo y transmitir contenido multimedia.

En la siguiente memoria se detalla la forma en la que se ha realizado la creación de un entorno web encargado de visualización de cámaras en directo, así como la infraestructura de red que se utiliza y su configuración.

1.1. Módulos a los que implica

Para llevar a cabo este proyecto he aplicado los conocimientos que he ido aprendiendo a lo largo del Ciclo Formativo de Grado Superior, ASIR, en los siguientes módulos:(CAMBIAR)

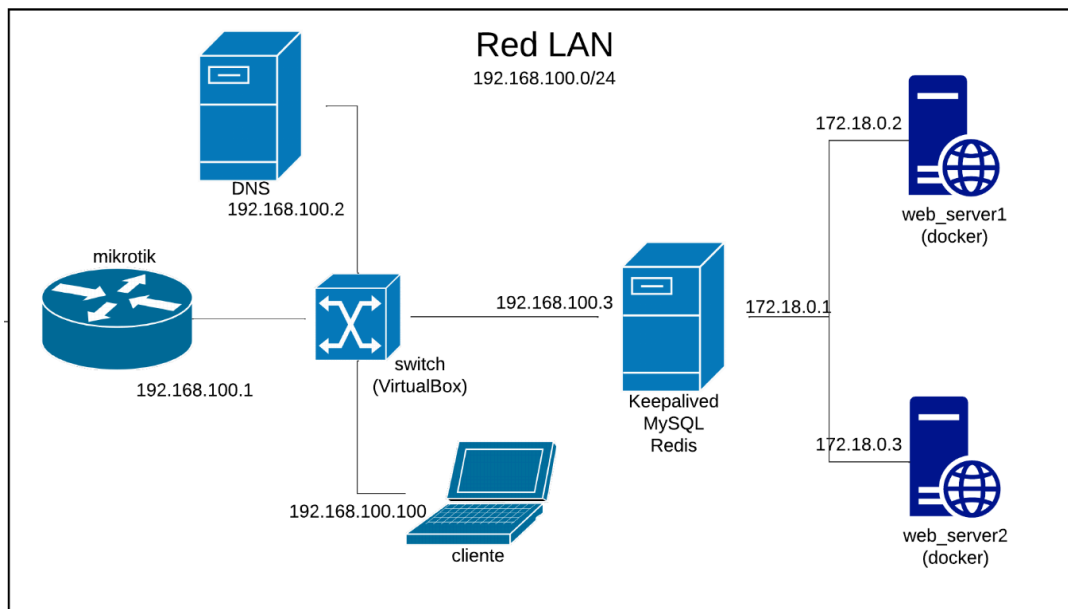
- Planificación y administración de redes: Integración red, SSH,, LAN, WAN, IPv4, acceso a internet
- Gestión de bbdd: Bbdd - Diagrama entidad-relación, DDL, DML, DCL
- Lenguajes de marcas: HTML, CSS, js
- Servicios de red e internet: Servicio DNS, DHCP, servidor web, servicios de audio, video...
- Implantación de aplicaciones web: Despliegue app web + bbdd
- Seguridad y alta disponibilidad: cifrado, HA, seguridad perimetral, seguridad red

1.2. Breve descripción del proyecto

Este proyecto trata sobre la creación de una infraestructura de red con diferentes elementos para garantizar la seguridad y la escalabilidad donde implementar una herramienta nuestra propia herramienta web en la que podamos ver nuestras cámaras de seguridad de forma cómoda y en cualquier lugar.

- Objetivos y requisitos del proyecto
- Uso del proyecto
- Tecnologías investigadas

2. *Diseño de red*

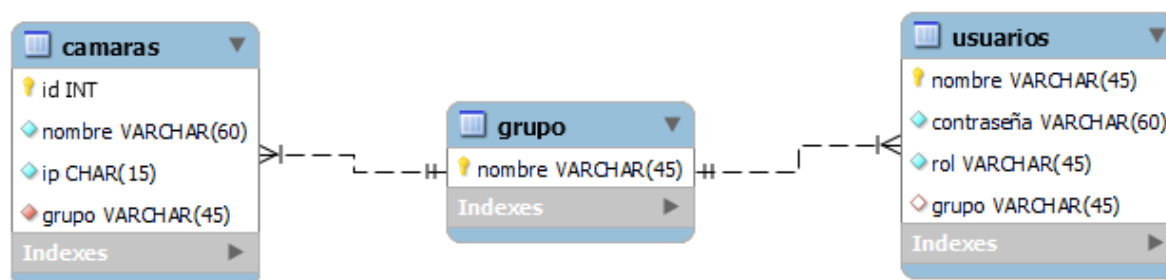


La estructura que garantiza la implementación exitosa y segura de los servicios, basada en el diseño de la red virtual, consiste en reunir con el debido cuidado los componentes esenciales. Los elementos son:

- **Router:** enruta redes externas utilizando mecanismos NAT que permiten controlar las conexiones entrantes y salientes.
- **Switch (Virtualbox):** garantiza una comunicación efectiva y confiable entre todos los dispositivos conectados a través de la red virtualizada.
- **Servidor DNS:** garantiza una conectividad perfecta para los usuarios que escriben URL en sus navegadores.
- **El servidor maestro:** implementa dos contenedores Docker como servidores web. Además, el servidor cumple otras funciones, como el balanceo de carga, almacenamiento de carpetas del sitio web, gestión de bases de datos y mantenimiento de variables de sesión PHP.

- Un ordenador que actúa como plataforma de virtualización y dos cámaras IP conectadas a la red externa son otros recursos de este entorno que dotan al sistema de nuevas capacidades y funciones.

3. *Base de datos*



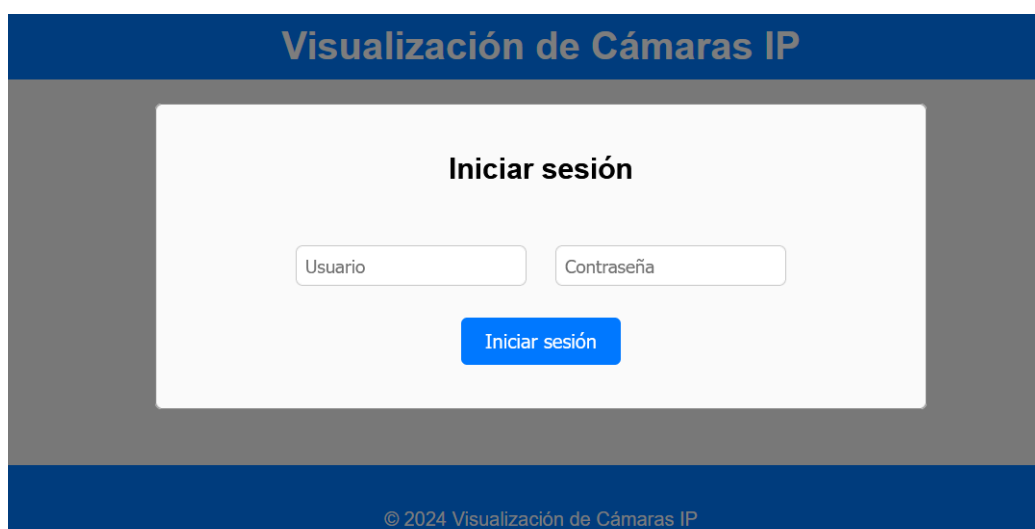
En este sistema, los usuarios pueden estar asociados a un grupo y un grupo puede contener múltiples usuarios y varias cámaras. Cada cámara solo puede ser parte de un único grupo.

Existen dos roles de usuario: Administrador y "viewer". Los administradores tienen privilegios para crear y administrar grupos y usuarios, así como visualizar todas las cámaras. Además, pueden añadir o eliminar cámaras en cualquier grupo. Los usuarios con el rol de "viewer" solo tienen permiso para añadir o eliminar cámaras en el grupo al que pertenecen, sin la capacidad de administrar grupos o usuarios.

4. WEB

Para este proyecto, se ha desarrollado esta página web donde poder visualizar nuestras cámaras, la idea es desplegar esta en los servidores de cada empresa/persona y así solo empleados/dueños de estos podrán visualizar sus cámaras de forma segura ya que no estará pública a internet.

Al intentar acceder a la página, vemos un menú donde si o si tendremos que iniciar sesión con nuestro usuario, sino no hay forma de entrar.



The image shows a web application interface for 'Visualización de Cámaras IP'. It features a dark blue header with the title 'Visualización de Cámaras IP' in white. Below the header is a light gray background with a white rectangular box in the center. Inside this box, the text 'Iniciar sesión' is displayed in bold. Below the text are two input fields: 'Usuario' and 'Contraseña'. A blue button with the text 'Iniciar sesión' is positioned below the input fields. At the bottom of the page, there is a dark blue footer with the text '© 2024 Visualización de Cámaras IP' in white.

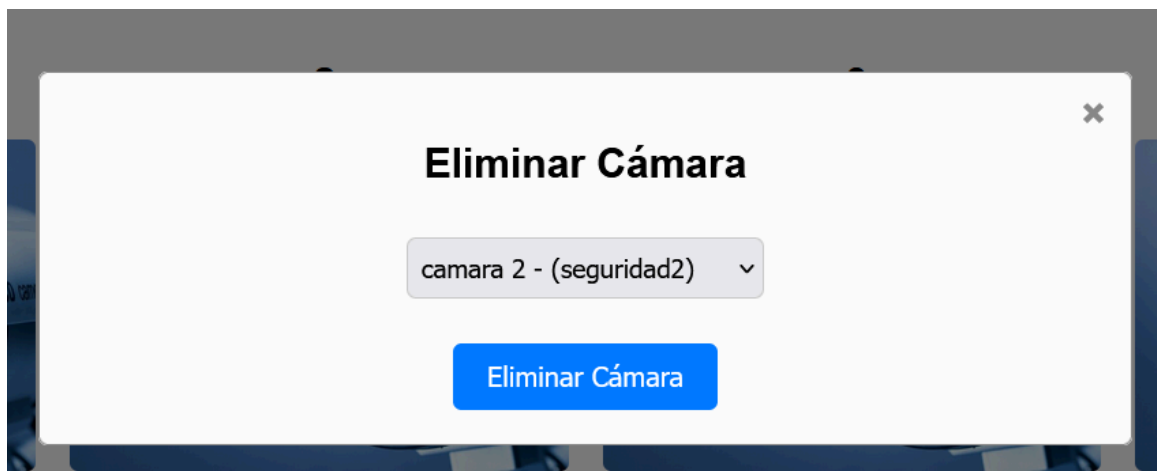
Una vez iniciada la sesión, (esta vez como administradores) podemos ver todas las cámaras registradas de cada grupo que hayamos creado. Estas tienen un nombre y un indicador del grupo al que pertenecen.



Si accedemos al botón “Añadir cámara” de la parte superior izquierda, veremos una ventana como la siguiente donde nos pide el nombre que queramos poner a la cámara, el grupo y los datos para poder verla.



Del mismo modo, en “Eliminar cámara” nos mostrará otra ventana donde podemos seleccionar la cámara que queremos eliminar.



En el caso de que seamos Administradores, tendremos un menú arriba a la derecha llamado usuarios, donde se podrá crear y eliminar grupos y añadir ó borrar usuarios, cambiar contraseña, cambiarlos de grupo y hacerlos administradores o no.



Si nos fijamos, cuando estamos en la página principal, a la izquierda del botón de usuarios, encontramos otro botón “cuenta” donde podremos apreciar los datos de esta, y cambiar la contraseña a nuestro gusto.



Por último, tenemos el botón de cerrar sesión donde se elimina la sesión actual para dar más seguridad y así si alguien consigue acceder a nuestro ordenador no pueda acceder a nuestras cámaras.

Una observación a tener en cuenta: el intento de acceder a cualquier página sin iniciar sesión correctamente le llevará directamente a la página de inicio de sesión. De manera similar, cualquier persona que no sea administrador y que intente acceder a la página de los usuarios será devuelta a la página de inicio o de inicio de sesión.

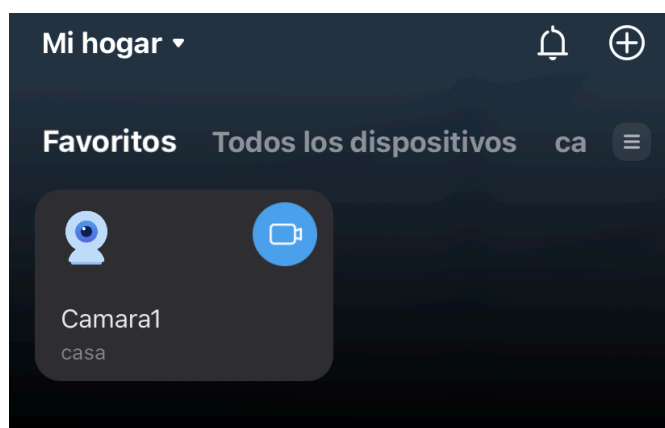
Existen restricciones para la eliminación de grupos cuando tienen cámaras o usuarios asociados para garantizar la integridad de los datos y la coherencia del sistema; sólo se pueden eliminar grupos vacíos.

Respecto al manejo de errores: confundir el nombre de inicio de sesión o la contraseña genera un mensaje de error de aclaración. De manera similar, las discrepancias en la confirmación de la contraseña dan como resultado un mensaje de error que solicita entradas precisas, evitando así cualquier confusión.

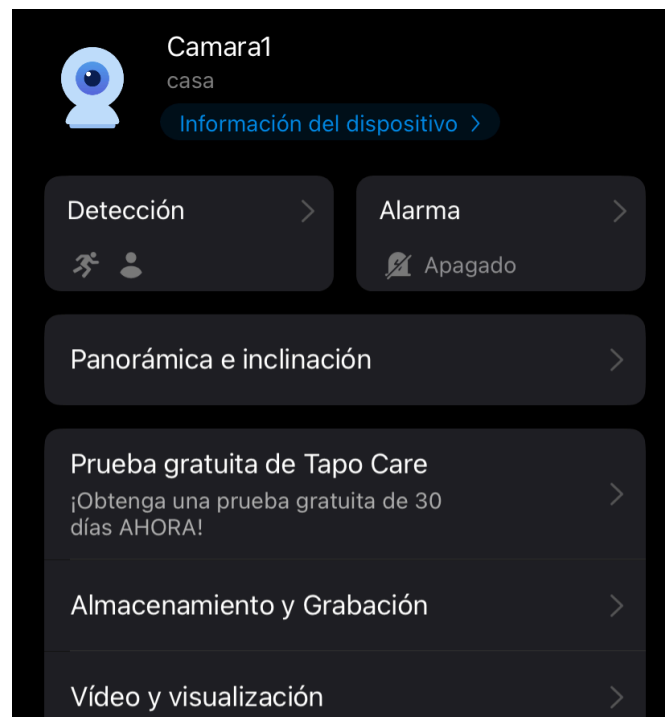
5. Configuración de cámaras

Para este proyecto, se implementó la conexión mediante el protocolo RTSP para la transmisión de vídeo en directo. Para comenzar, las cámaras, en este caso las Tapo C200, deben estar conectadas a la red local. Para lograr esto, se utiliza un dispositivo móvil para asociar las cámaras al router y se crea una cuenta que permite acceder al RTSP.

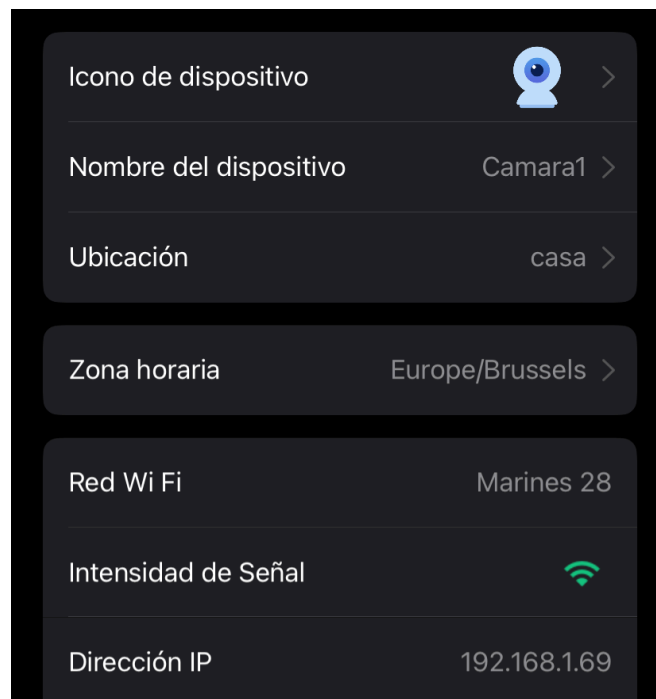
Al añadirlas a la app a través del del wifi indicándoles la contraseña y el nombre que le queremos asignar, podemos entrar en la configuración de la cámara pulsando sobre ella,



Dentro, podemos encontrar opciones de configuración para su funcionamiento como activar la alarma, opciones de detección de intrusos con notificación, etc.

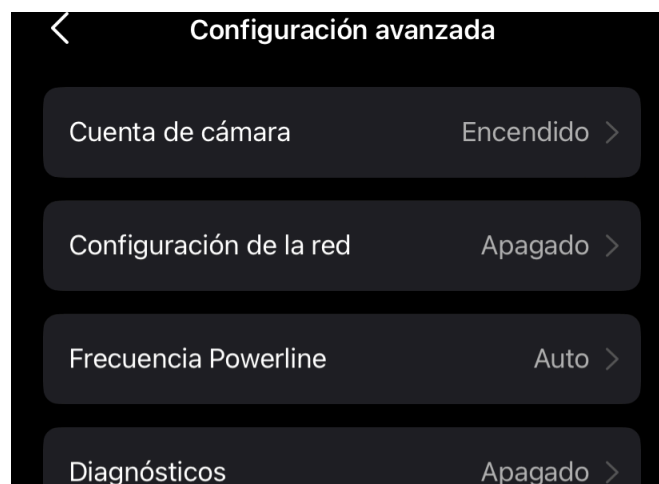


Si pulsamos en “Información del dispositivo”, podemos ver diferente información como: la red wifi asociada, el nombre de la cámara, zona horaria y entre otras cosas la IP, que es lo que más nos importa en este momento.



Además de la IP, para poder introducir la cámara en nuestro sistema, necesitamos crear una cuenta para poder acceder a ella, por lo general será en todas las cámaras la misma, el mismo usuario y contraseña.

Esta cuenta se puede crear en el apartado “Cuenta de cámara” dentro de la configuración avanzada que encontramos en la pestaña principal de configuración de cada dispositivo.



Una vez realizados estos pasos, se pueden agregar a la interfaz web. Por defecto, estas cámaras utilizan el puerto 554 para la transmisión de video, por lo que la URL de acceso a cada cámara sería del tipo:

`rtsp://usuario.contraseña@ip-de-la-camara:554/stream1`

Lamentablemente, los navegadores web no son compatibles con este formato de video. Por tanto, existen varias alternativas para visualizar el contenido:

- **Protocolo ONVIF:** Se podría utilizar este protocolo para conectarse a las cámaras, pero requeriría el uso de un cliente externo compatible con ONVIF o desarrollar un cliente propio, lo cual podría resultar complicado.
- **Conversión de vídeo:** Otra opción sería crear una máquina que convierta el flujo de video de RTSP a formatos compatibles con los navegadores, como RTMP o HTTP. Sin embargo, esto podría no ser viable debido a la falta de recursos y a la complejidad técnica que implica.
- **Visualizadores de video:** La alternativa más práctica sería utilizar visualizadores de video como VLC, ffmpeg o mplayer. De esta manera, se mantendría la misma estructura en la interfaz web, pero al hacer clic en una cámara, se abriría una ventana emergente del programa seleccionado con el video en directo.

Finalmente, se optó por la tercera opción. Esto se traduce a: al hacer clic en una cámara, se abre una ventana flotante que muestra el video en tiempo real utilizando el visualizador de video elegido previamente.

Esta elección se basó en su practicidad y facilidad de implementación. Evita la complejidad técnica asociada con la conversión de formatos de video o el desarrollo de clientes personalizados, lo que la convierte en una opción práctica y eficiente.

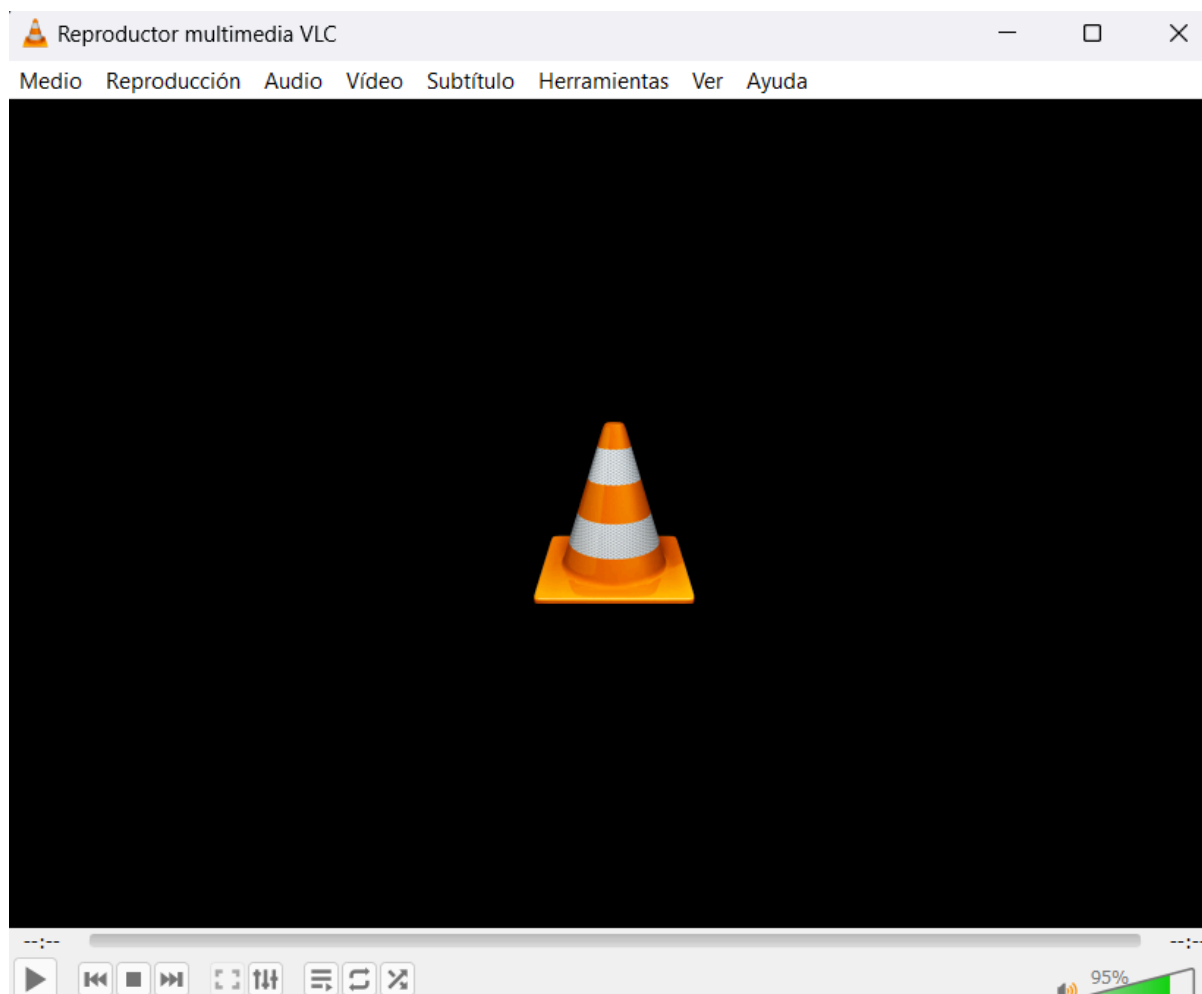
5.1. Integración de cámaras en la web

Para poder visualizar todas las cámaras que introducimos en la web, he implementado un bucle después de iniciar una conexión con la base de datos donde se crea cada elemento de cámara con el enlace correspondiente para su visualización a través de la ip que tenga asociada

```
// Consulta a la base de datos
if(isset($_SESSION["rol"]) && $_SESSION["rol"]=="admin" ) {
    $sql = "SELECT * FROM camaras order by grupo ";
} else {
    $grupo = $_SESSION["grupo"];
    $sql = "SELECT * FROM proyecto.camaras where grupo = '$grupo'";
}

$resultado = $conexion->query($sql);

// Procesar los resultados
if ($resultado->num_rows > 0) {
    while ($fila = $resultado->fetch_assoc()) {
        ?>
        <div class="camera">
            <h2>
            <?php echo $fila['nombre']; ?>
            <?php if(isset($_SESSION["rol"]) && $_SESSION["rol"]=="admin" ) {
?>
                <span class="rol"><?php echo '($_SESSION["rol"]); ?></span>
                <?php } ?>
            </h2>
            "
class="portada" onclick="mostrarVideo('<?php echo $fila['url']; ?>')">
            </div>
            <?php
        }
    }
}
```



6. Configuración Servidor 1

6.1. Tarjetas de red

Ya que este servidor será el principal, es crucial que mantenga una dirección IP fija. Por lo tanto, procederemos a configurarla manualmente en el archivo **“/etc/netplan/00-installer-config.yaml”** de la siguiente manera, definiendo la IP como 192.168.100.2.

```
network:
  ethernets:
    enp0s10:
      dhcp4: true
    enp0s3:
      dhcp4: false
      addresses: [192.168.100.2/24]
      gateway4: 192.168.100.1
      nameservers:
        addresses: [192.168.100.3, 8.8.8.8, 8.8.4.4]
  version: 2
```

6.2. Configuración Apache2

Ahora, vamos a crear el archivo de virtual host para cada servidor web que levantaremos más adelante.

Este archivo lo introduciremos en cada servidor web en la carpeta **“/etc/apache2/sites-available/”** con el nombre **“default-ssl.conf”** reemplazando el archivo por defecto que se crea junto la instalación del apache.

```
<IfModule mod_ssl.c>
  <VirtualHost _default_:443>
    DocumentRoot /var/www/html
    Protocols h2 http/1.1

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
```

```

SSLEngine on
SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

<FilesMatch "\.(cgi|shtml|phtml|php)$">
SSLOptions +StdEnvVars
</FilesMatch>
<Directory /usr/lib/cgi-bin>
SSLOptions +StdEnvVars
</Directory>

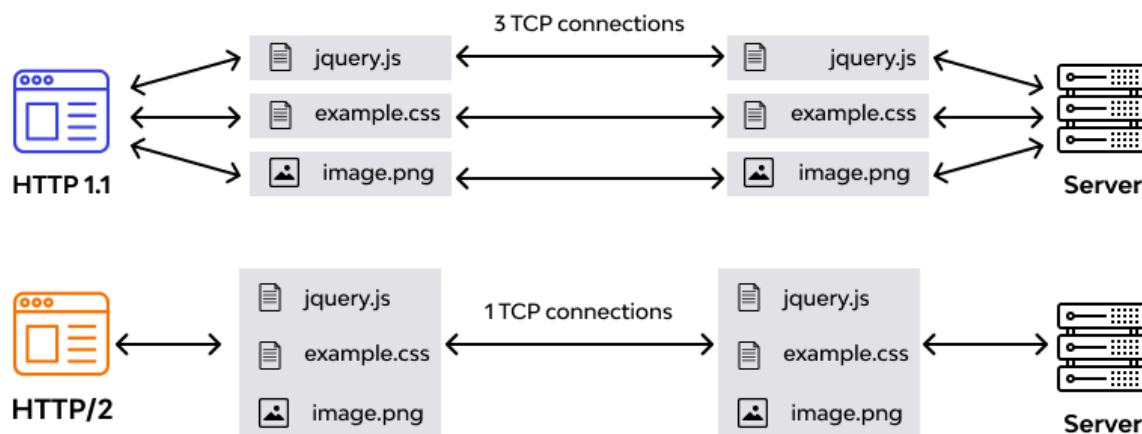
ProxyPassMatch                               ^/(.*\.php(/.*)?)$
unix:/var/run/php/php8.1-fpm.sock|fcgi://localhost/var/www/html/$1

</VirtualHost>
</IfModule>

```

En esta configuración, observe la apariencia del protocolo HTTP/2 denominado "**Protocols h2**"; supera con creces las capacidades de rendimiento del estándar HTTP/1 simplemente por su tipo de conexión: una observación de alto octanaje.

Para comprender visualmente este concepto:



Dirigiendo nuestra atención a la configuración SSL.

Observe esos certificados de "**snakeoil**", lejos de cualquier apariencia de autenticidad. Estos son certificados de prueba o generados automáticamente que el propio servidor crea.

Su propósito es crear un entorno simulado SSL sin tener que adquirir un certificado oficial. Un acto muy útil durante las pruebas de desarrollo de proyectos, donde dichas simulaciones pueden ser suficientes sin necesidad de validación en el mundo real.

Por último, hacia el final del archivo, verás unas líneas que mencionan "php8.1-fpm". Esta línea es fundamental para lograr dos tareas principales: separar el servicio PHP del módulo Apache mediante PHP-FPM (FastCGI Process Manager). Al hacerlo, el servidor puede manejar las operaciones de manera más eficiente ya que procesa PHP de forma independiente. Además, este cambio no compromete la compatibilidad con el protocolo http 2, a diferencia del módulo Apache PHP que carece de compatibilidad con esta característica.

6.3. Docker

Docker es más que una plataforma de contenedores, es una herramienta de empaquetado, distribución y ejecución para aplicaciones en contenedores que son portátiles y aislados. Los beneficios van desde la portabilidad obvia hasta el aislamiento, la eficiencia y la coherencia menos obvios en toda la implementación de la aplicación. Un Dockerfile prepara el escenario para la construcción de contenedores: define qué entorno se debe configurar junto con qué dependencias se deben agregar, incluso hasta los comandos al inicio.

6.3.1. Dockerfile

Este documento tiene dos propósitos: en primer lugar, detallar dónde se puede encontrar el Dockerfile (/home/victor/dockerfile/dockerfile en el Servidor 1) y, en segundo lugar, establecer qué configuraciones de entorno específicas se detallan en

este archivo para garantizar una implementación y operación efectivas. del servicio del proyecto.

```
-----
FROM ubuntu:latest

ENV DEBIAN_FRONTEND=noninteractive
ENV TZ=Europe/Berlin

RUN apt-get update && \
    apt-get install -y apache2 php8.1 php8.1-fpm php8.1-mysql php8.1-redis \
    iproute2 nano && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

RUN a2dissite 000-default.conf
RUN a2ensite default-ssl.conf

RUN a2enmod ssl rewrite proxy_fcgi mpm_event http2 && \
    a2dismod mpm_prefork

RUN echo "session.save_handler = redis" >> /etc/php/8.1/fpm/php.ini \
    && echo "session.save_path = tcp://172.18.0.1:6379" >> \
    /etc/php/8.1/fpm/php.ini

COPY default-ssl.conf /etc/apache2/sites-available/default-ssl.conf

CMD ["sh", "-c", "service php8.1-fpm start && apachectl -D FOREGROUND"]
-----
```

- **Imagen:** Ubuntu 22.04.4 LTS. Seleccione esta versión de la imagen para asegurar su estabilidad y compatibilidad con los paquetes y servicios necesarios.
- **Variables de Entorno:**
 - “**DEBIAN_FRONTEND**” está configurada como no interactiva, lo que permite el proceso de instalación del paquete sin que sea necesaria la intervención del usuario.
 - “**TZ**” es un ajuste de zona horaria a Europa/Berlín, lo que hace posible que todos los servicios dependiendo de la hora funcionen correctamente y los registros de eventos se sincronicen.

- **Paquetes de instalación:**

- **Apache:** Un potente servidor web muy utilizado.
- **PHP 8.1:** Un lenguaje de programación para desarrollo web, junto con sus módulos que facilitan la integración con MySQL y Redis.
- **Utilidades adicionales:** iproute2 (usada para gestión de red) y nano (como editor de texto que ayuda en la administración dentro del contenedor).

Después de la instalación, se realiza una limpieza de caché adecuada; esto reduce el tamaño de la imagen.

- **Configuración de Apache:** El sitio predeterminado está deshabilitado y una configuración SSL personalizada está habilitada. Además, ciertos módulos (rewrite, proxy_fcgi, mpm_event, http2) están activados mientras que otros (mpm_prefork) están deshabilitados, para mejorar el rendimiento y la compatibilidad con las tecnologías utilizadas.
- **Configuración de PHP:** PHP se configura para utilizar Redis como almacenamiento de sesiones implica dos pasos principales: definir Redis como controlador de sesiones y especificar la ruta de conexión al servicio de Redis. Esta configuración particular mejora la gestión de sesiones de la aplicación, lo que a su vez aumenta la eficiencia y escalabilidad de la aplicación.
- **Copiado de Archivos de Configuración:** Se agrega un archivo de configuración SSL personalizado y luego se copia en el directorio de sitios disponibles de Apache. Esto garantiza que el servidor web funcione con requisitos específicos de seguridad y rendimiento que se hayan implementado.
- **Comando Predeterminado del Contenedor:** Al iniciar el contenedor, se especifica un comando para iniciar PHP-FPM y el servicio Apache en modo

de primer plano. Al hacer esto, ambos servicios estarán activos y preparados para manejar solicitudes inmediatamente después del lanzamiento del contenedor.

6.3.2. Docker compose

El archivo `docker-compose.yml`, también ubicado en el servidor 1, en `/home/victor/dockerfile/docker-compose`, define la estructura de la aplicación a través de servicios y especifica la configuración del volumen para mantener la sincronización de archivos entre el servidor y los contenedores Dockers.

```
-----  
version: '3.8'  
  
services:  
  web_server1:  
    build:  
      context: .  
      dockerfile: dockerfile  
    volumes:  
      - /home/victor/proyecto/web:/var/www/html  
  web_server2:  
    build:  
      context: .  
      dockerfile: dockerfile  
    volumes:  
      - /home/victor/proyecto/web:/var/www/html  
-----
```

Aquí definimos dos servicios en este ejemplo: `web_server1` y `web_server2`, que representan dos instancias del servidor web. Cada servicio está configurado para utilizar el mismo Dockerfile que se especifica en el directorio actual (`context: .` y `dockerfile: dockerfile`).

La sección de volúmenes explica cómo se gestionan los volúmenes dentro de los contenedores. En nuestro caso, montamos el directorio `/home/victor/project/web` del host en `/var/www/html` dentro de los contenedores. Esta sincronización bidireccional (host a contenedores, contenedores a host) garantiza que todos los contenedores

tengan acceso uniforme a los archivos de aplicaciones actualizados. Se establece una paridad entre los entornos de desarrollo y producción, lo que facilita el desarrollo y la depuración al permitir coherencia entre diferentes entornos.

6.4. Servidor de variables de sesion php

Redis es una base de datos en memoria de código abierto que se utiliza como un servidor aparte para almacenar datos críticos o temporales que son necesarios para mantener el estado de la aplicación web. A menudo se emplea para almacenar variables de sesión de PHP u otros lenguajes de programación, lo que permite que múltiples instancias del servidor web compartan y accedan a estas variables de manera consistente.

El servidor de variables de sesión PHP utiliza Redis como servidor aparte para almacenar estas variables, lo que asegura la consistencia y disponibilidad de las sesiones, especialmente cuando el balanceador de carga distribuye las solicitudes entre varios servidores web.

El archivo de configuración de Redis, ubicado en `/etc/redis/redis.conf`, se ajusta para especificar las direcciones IP en las que el servidor debe escuchar las conexiones entrantes:

```
-----  
bind 127.0.0.1 172.18.0.1 ::1  
-----
```

Esta configuración indica al servidor Redis que escuche las conexiones en las direcciones IP 127.0.0.1, **172.18.0.1** y `::1`. La inclusión de 172.18.0.1 permite que los contenedores Docker, que pueden tener esta dirección asignada por el puente de red interno, se comuniquen con el servidor Redis para el almacenamiento de las variables de sesión PHP. Esto asegura que las variables de sesión se mantengan consistentes y accesibles, independientemente de en qué servidor web se procesen

las solicitudes, evitando problemas de sesión al cambiar entre diferentes instancias del servidor web a través del balanceador de carga.

6.5. Balanceador de carga

Keepalived es una herramienta de servicios de alta disponibilidad de código abierto para Linux y balanceador de carga. Ayuda a garantizar que se logre la continuidad del servicio mediante la detección de fallas de los servidores primarios, con conmutación automática entre servidores redundantes, minimizando así el tiempo de inactividad. Además, también dirige el tráfico de la red a diferentes servidores, lo que también mejora el rendimiento y la confiabilidad del sistema.

Existe un requisito para habilitar el reenvío de paquetes entre interfaces porque los servidores web están ubicados en una red privada. Esta tarea se puede realizar agregando como usuario root la línea “net.ipv4.ip_forward = 1” al archivo “/etc/sysctl.conf”.

Haremos uso de esto ejecutando sysctl -p.

```
victor@victor-server:~$ sudo nano /etc/sysctl.conf
victor@victor-server:~$ sudo sysctl -p
net.ipv4.ip_forward = 1
victor@victor-server:~$ |
```

Al adquirirlo identificamos el nombre de la interfaz pública y realizamos la posterior regla iptables que dirige el tráfico de la red interna a la externa.

```
victor@victor-server:~$ ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:d7:81:a5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.3/24 brd 192.168.100.255 scope global enp0s3
        valid_lft forever preferred_lft forever
```

```
sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
```

```
victor@victor-server:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination
DOCKER     all  --  anywhere                anywhere        ADDRTYPE match dst-type LOCAL

Chain INPUT (policy ACCEPT)
target     prot opt source                destination
DOCKER     all  --  anywhere                !localhost/8    ADDRTYPE match dst-type LOCAL

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
DOCKER     all  --  anywhere                !localhost/8    ADDRTYPE match dst-type LOCAL

Chain POSTROUTING (policy ACCEPT)
target     prot opt source                destination
MASQUERADE all  --  172.18.0.0/16          anywhere
```

Esto nos lleva a la configuración particular del balanceo de carga, que se encuentra en el archivo `/etc/keepalived/keepalived.conf` con las reglas de balanceo para el servidor principal.

```
global_defs {
    router_id LVS_PRACTICA
}

virtual_server 192.168.103.3 443{
    delay_loop 3
    lb_algo rr
    lb_kind NAT
    persistence_timeout 0
    protocol TCP

    real_server 172.18.0.2 443{      #docker1
        weight 1
```

```
TCP_CHECK {  
    connect_port 443  
    connect_timeout 3  
}  
}  
  
real_server 172.18.0.3 443{      #docker2  
    weight 1  
    TCP_CHECK {  
        connect_port 443  
        connect_timeout 3  
    }  
}  
}
```

Con esta configuración, tenemos el servidor principal en la dirección IP 192.168.103.3 en el puerto 443, utilizando el algoritmo de equilibrio por turnos (lb_algo rr) combinado con el tipo de equilibrio NAT (lb_kind NAT); persistencia de conexión deshabilitada y protocolo configurado como TCP (persistence_timeout 0).

Así, tenemos definidos los dockers, docker1 y docker2, con sus direcciones IP 172.18.0.2 y 172.18.0.3, ambos ejecutándose en el puerto 443, con sus estados confirmados individualmente a través de una conexión TCP al puerto 443 (tiempo de espera de 3 segundos).

7. Configuración servidor 2

7.1. tarjetas de red

Ya que este servidor es el servidor DNS, es necesario asignarle una dirección IP fija al igual que hicimos con el otro servidor para garantizar su estabilidad y accesibilidad en la red. Para lograr esto, configuraremos manualmente la dirección IP en el archivo "/etc/netplan/00-installer-config.yaml", asignándole la dirección IP 192.168.100.2.

```
network:
```

```

ethernets:
  enp0s10:
    dhcp4: true
  enp0s3:
    dhcp4: false
  addresses: [192.168.100.2/24]
  gateway4: 192.168.100.1
  nameservers:
    addresses: [192.168.100.2, 8.8.8.8, 8.8.4.4]
version: 2

```

7.2. Configuración DNS

Para la configuración del DNS con el paquete Bind9, se ha creado una zona en el archivo "named.conf.local" con el nombre "securewatch.com" como nombre de dominio para nuestra web. La zona tiene la siguiente estructura:

```

zone "securewatch.com" {
    type master;
    file "/etc/bind/db.securewatch.com";
};

```

Con la zona ya definida, creamos su archivo de zona "**/etc/bind/db.securewatch.com**" donde declaramos los registros de resolución para nuestro dominio "securewatch.com". Estos registros incluyen información importante que ayuda a dirigir el tráfico de Internet a los recursos correctos asociados a nuestro dominio.

```

db.securewatch.com ;
; BIND data file for local loopback interface
;
$TTL 604800
@           IN      SOA  ns1. root.localhost. (
                        2      ; Serial
                        604800 ; Refresh
                        86400  ; Retry
                        2419200 ; Expire
                        604800 ) ; Negative Cache TTL
;

```

securewatch.com.	IN	NS	ns1.
ns1	IN	A	192.168.100.2
www	IN	A	192.168.100.3

8. Mikrotik

8.1. ¿Qué es un router Mikrotik?

Un router MikroTik es un dispositivo de red que forma parte de la amplia gama de soluciones ofrecidas por la empresa MikroTik. Estos routers se destacan por su versatilidad, potencia y capacidad para gestionar tanto redes de área local (LAN) como redes de área amplia (WAN). En su núcleo, un router MikroTik está equipado con el sistema operativo RouterOS, desarrollado por MikroTik, que brinda un conjunto completo de características y funcionalidades avanzadas para la gestión de redes. Estos dispositivos son altamente reconocidos por su capacidad para realizar enrutamiento estático y dinámico, configuración de firewall, gestión de ancho de banda, y una variedad de servicios de red.

8.2. Instalación y configuración inicial

Descargamos la ova de su página oficial:

<https://download.mikrotik.com/routeros/7.11.2/chr-7.11.2.ova>

Le configuramos una tarjeta de red en NAT o puente, y otra en ip interna que será la red que queremos crear para esta infraestructura.

Al arrancar la máquina (usuario admin, contraseña en blanco) podemos acceder a la ip “publica” desde nuestro navegador para empezar a configurarlo.

MikroTik

Safe Mode Quick Set WebFig Terminal

Ethernet Quick Set

Mode ☒ Router ☐ Bridge

▼ Internet

Address Acquisition ☐ Static ☒ Automatic ☐ PPPoE

IP Address 10.0.2.15 Renew Release

Netmask 255.255.255.0 (/24)

Gateway 10.0.2.2

MAC Address 08:00:27:85:F2:5A

▼ Local Network

IP Address 192.168.100.1

Netmask 255.255.255.0 (/24) ▼

Bridge All LAN Ports ☐

DHCP Server ☒

DHCP Server Range ▲ 192.168.100.1-192.168.100.200

NAT ☒

Como vemos, asignamos la ip del router de nuestra LAN (192.168.100.1), asignamos un pool de DHCP para los clientes que se conecten a esta red. Activamos la opción NAT para poder realizar el nateo y permitir el acceso a internet de los equipos que están en la LAN.

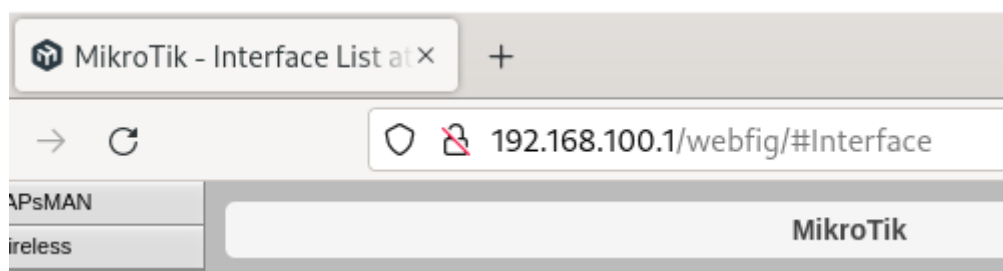
ALARGAR

8.3. Montar VPN

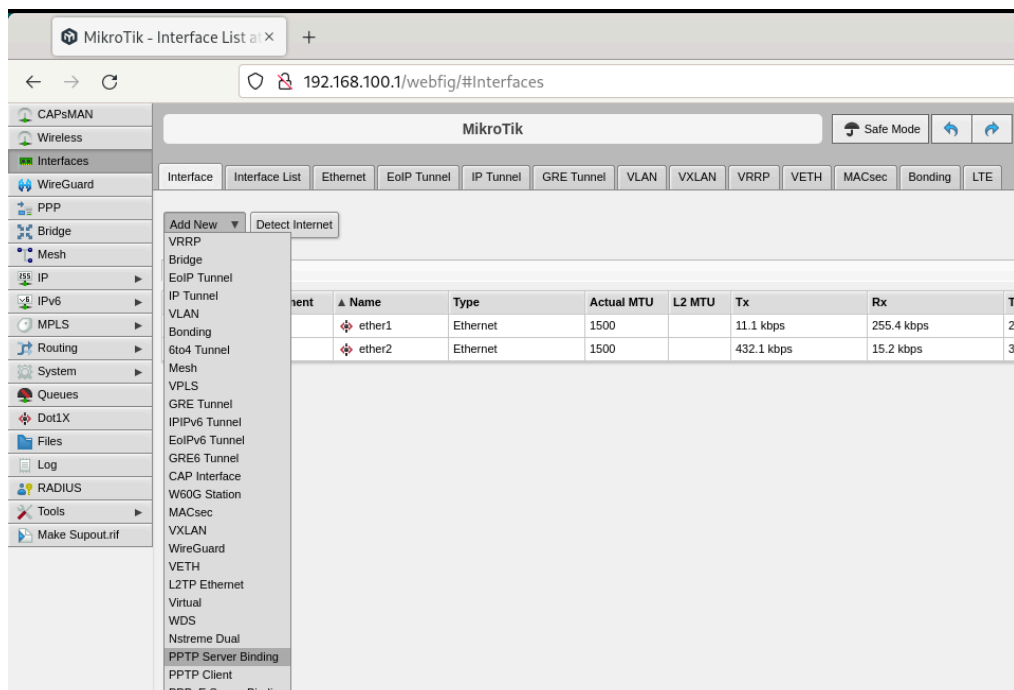
Accedemos a la máquina Mikrotik en modo consola para poder ver la dirección ip pública que nos ha asignado. Cuando digo pública es aquella que nos asigna al adaptador en modo puente.

```
[admin@MikroTik] > ip address/print
Flags: D - DYNAMIC
Columns: ADDRESS, NETWORK, INTERFACE
# ADDRESS NETWORK INTERFACE
0 192.168.100.1/24 192.168.100.0 ether2
1 192.168.15.8/24 192.168.15.0 ether1
```

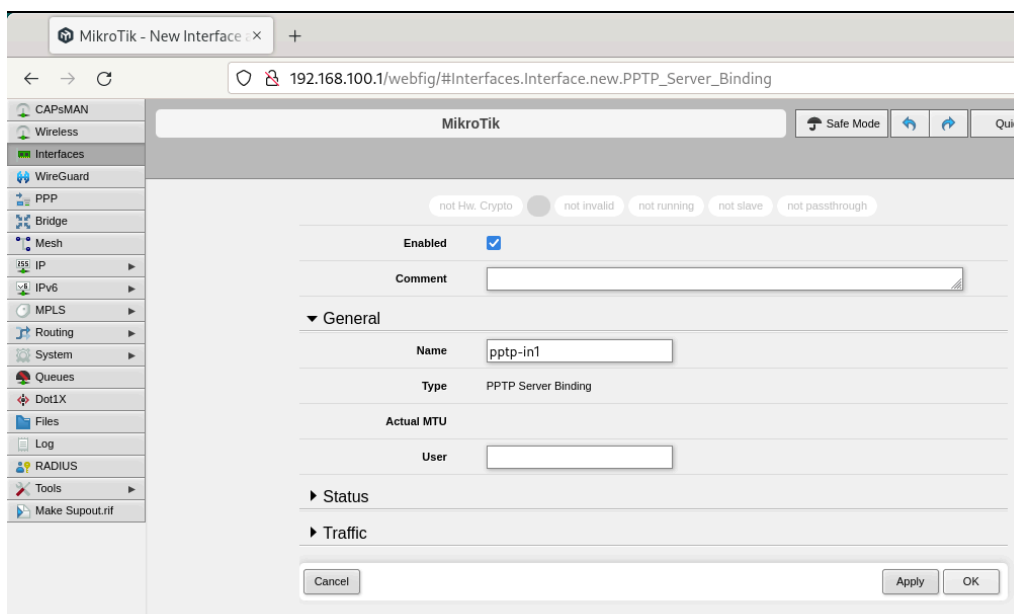
Con esa IP accedemos a través del interfaz web al router Mikrotik:

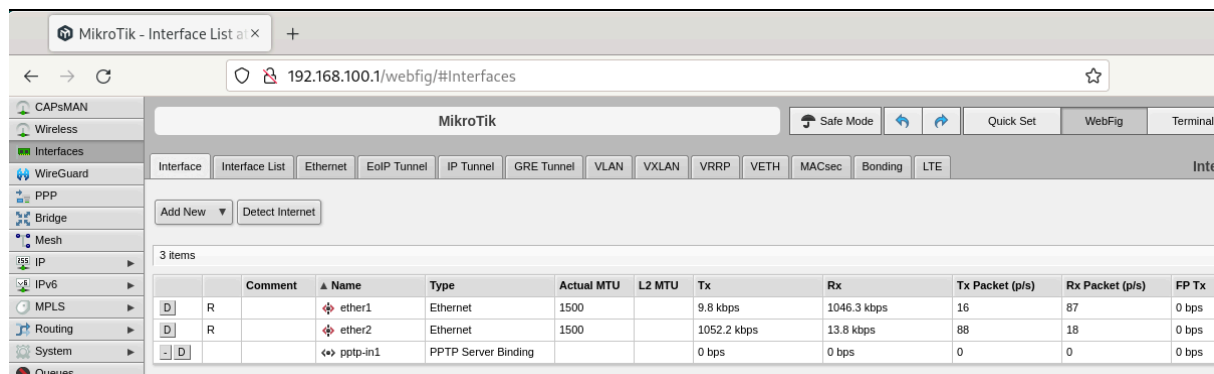


Una vez estamos en este vamos a la opción WebConfig y pulsamos en la parte izquierda sobre interfaces. En esta añadimos (Add New) un nuevo interfaz de tipo PPTP Server Binding



Aquí sólo le damos a ok y listo:

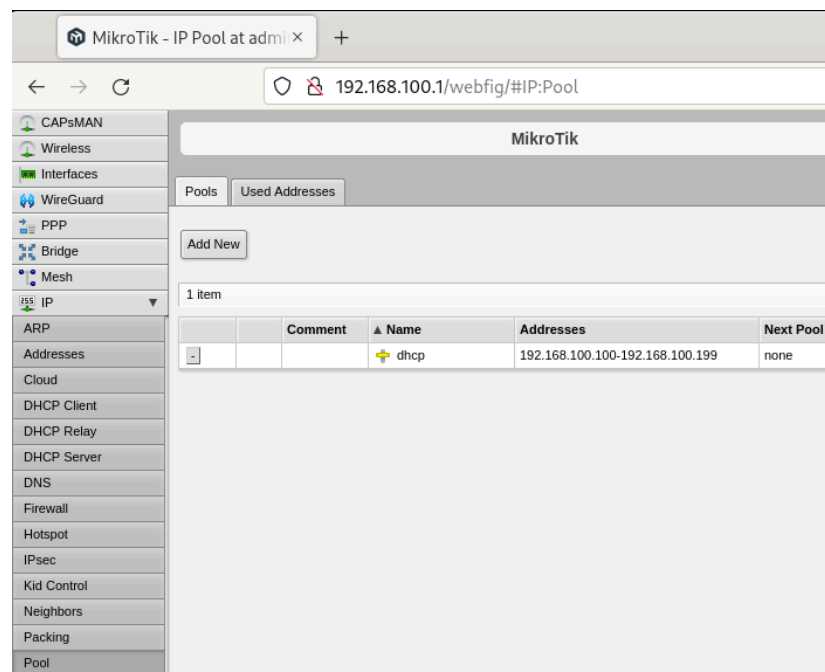




The screenshot shows the MikroTik WebFig interface for the 'Interfaces' section. The left sidebar contains navigation links for CAPsMAN, Wireless, Interfaces, WireGuard, PPP, Bridge, Mesh, IP, IPv6, MPLS, Routing, System, and Queues. The main content area displays a table of 3 items:

	Comment	Name	Type	Actual MTU	L2 MTU	Tx	Rx	Tx Packet (p/s)	Rx Packet (p/s)	FP Tx
[D]	R	ether1	Ethernet	1500		9.8 kbps	1046.3 kbps	16	87	0 bps
[D]	R	ether2	Ethernet	1500		1052.2 kbps	13.8 kbps	88	18	0 bps
[D]		pptp-in1	PPTP Server Binding			0 bps	0 bps	0	0	0 bps

El siguiente paso que vamos a realizar es ir a la sección IP y dentro de esta pulsamos en Pool



The screenshot shows the MikroTik WebFig interface for the 'IP Pool' section. The left sidebar is expanded to show the 'IP' section, with sub-links for ARP, Addresses, Cloud, DHCP Client, DHCP Relay, DHCP Server, DNS, Firewall, Hotspot, IPsec, Kid Control, Neighbors, Packing, and Pool. The main content area displays a table of 1 item:

	Comment	Name	Addresses	Next Pool
[+]		dhcp	192.168.100.100-192.168.100.199	none

Aquí como podemos ver tenemos configurado el pool para las direcciones internas que asigna el router MikroTik para las direcciones IP internas.

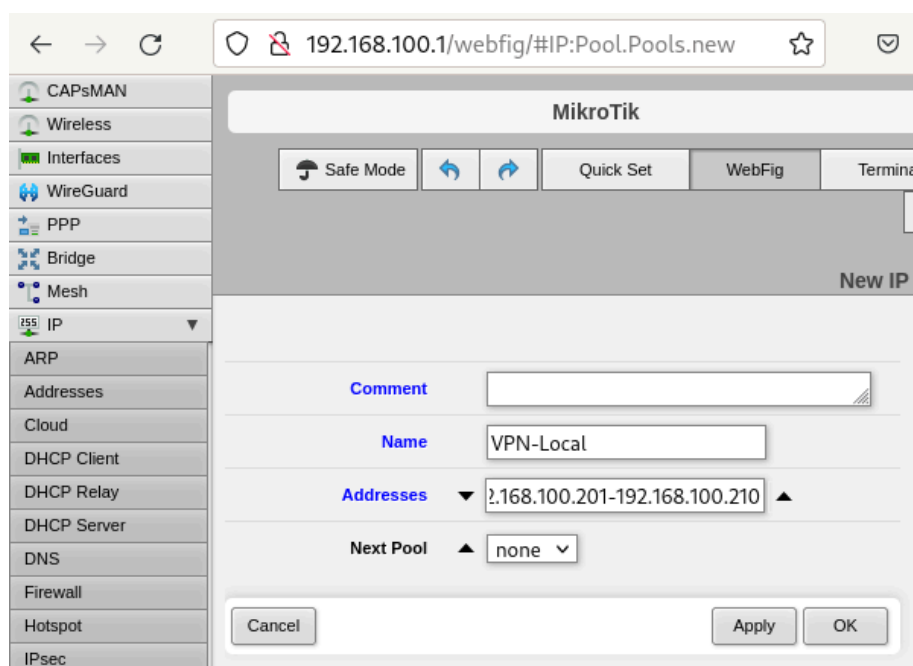
Lo que vamos a hacer es configurar también el pool de direcciones para aquellos que se conectan por VPN.

Lo podemos hacer de forma estática o dinámica, pero de forma estática deberemos ir usuario a usuario en la sección “secret” dando a cada usuario la ip local y la ip remota.

Para ello procederemos a crear un pool de asignación. Pulsamos en Add new.

Aunque no se vea en la captura de bajo el nombre (Name) que le doy es VPN-Local y el rango que direcciones IP que vamos a asignar (Addresses) para que no colisionen con las de DHCP son las siguientes:

192.168.100.201-192.168.100.210



Le pulsamos OK.

Y podemos ver los dos pools:

Pools

Used Addresses

IP Pool

Add New

2 items

		Comment	▲ Name	Addresses
-			dhcph	192.168.100.100-192.168.100.199
-			VPN-Local	192.168.100.201-192.168.100.210

Ahora siguiendo el mismo procedimiento vamos a añadir otro pool, que llamaremos VPN-Remoto, que irá de la 192.168.100.211 a la 192.168.100.220.

Add New					
3 items					
		Comment	▲ Name	Addresses	
-			dhcp	192.168.100.100-192.168.100.199	
-			VPN-Local	192.168.100.201-192.168.100.210	
-			VPN-Remoto	192.168.100.211-192.168.100.220	

Ahora vamos a la sección PPP que vemos en la imagen de la parte izquierda de la captura anterior.

Aquí podemos ver que la tarjeta virtual que hemos creado en interfaces está visible en esta sección:

←

→

↺

192.168.100.1/webfig/#PPP

CAPsMAN

Wireless

Interfaces

WireGuard

PPP

Bridge

Mesh

IP

ARP

Addresses

MikroTik

Safe Mode

↺

↻

Interface

PPPoE Servers

Secrets

Profiles

Active Connections

L2TP Ethernet

L2TP Secrets

Add New ▼

PPP Scanner

PPTP Server

SSTP Server

L2TP Server

OVPN Server

Import .ovpn

PPPoE Scan

1 item

	Comment	▲ Name	Type	Actual MTU	L2 MTU	Tx
- D		↔ pptp-in1	PPTP Server Binding			0 bps

Ahora vamos a la sección de Profiles y vamos a crear un nuevo profile. Le pondremos el nombre de VPN-enc de encriptado.

De Local Addresses seleccionamos el pool de VPN-Local

De Remote Addresses seleccionamos VP-Remoto

PPP

- Bridge
- Mesh
- IP
- ARP
- Addresses
- Cloud
- DHCP Client
- DHCP Relay
- DHCP Server
- DNS

Comment

▼ General

Name: VPN-enc

Local Address: VPN-Local

Remote Address: VPN-Remoto

Remote IPv6 Prefix Pool

Por último marcamos que utilice encriptación marcando yes:

Use UPnP ☐ no ☒ yes ☐ default

Hacemos clic en Ok y vemos que tenemos el nuevo profile.

PPP

- Bridge
- Mesh
- IP
- ARP
- Addresses
- Cloud
- DHCP Client
- DHCP Relay
- DHCP Server
- DNS

Add New

3 items

	Comment	▼ Name	Local Address	Remote Address	Bridge	Rate Limit (rx/tx)	Only One
-		VPN-enc	VPN-Local	VPN-Remoto			default
-	*	default-encrypti					default
-	*	default					default

En el siguiente paso vamos a generar un nuevo usuario. Para ello pulsamos la pestaña de Secrets y opción Add New

En lugar de hacerlo por máquinas lo vamos a realizar por personas. Como nombre colócale el tuyo y como contraseña por ejemplo 54321.

Como servicio PPTP y cómo profile el que acabamos de crear (VPN-enc)

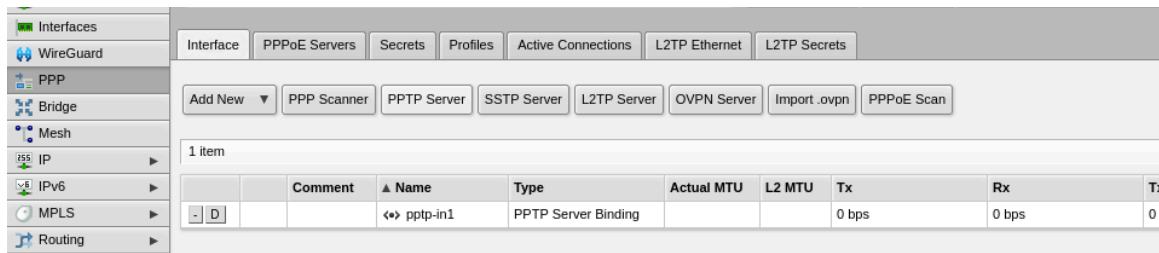
The screenshot shows the MikroTik WinBox interface for configuring a new PPP secret. The left sidebar lists various network services, with 'PPP' selected. The main panel shows the configuration for a new secret. The 'Enabled' checkbox is checked. The 'Name' field is 'victor', the 'Password' is masked with dots, the 'Service' is 'pptp', and the 'Profile' is 'VPN-enc'.

Pulsamos en Ok y comprobamos que ya lo tenemos:

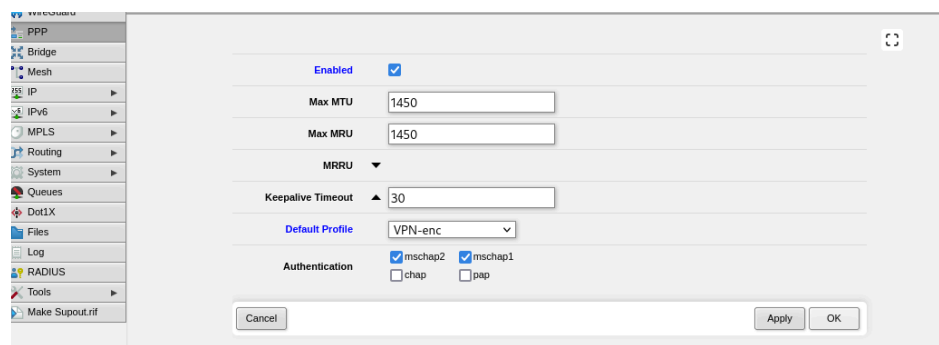
The screenshot shows the MikroTik WinBox interface for the list of configured PPP secrets. The table has columns for Comment, Name, Password, Service, Caller ID, Profile, Local Address, Remote Address, Routes, and Last Logged Out. One item is listed with Name 'Victor', Password '*****', Service 'pptp', and Profile 'VPN-enc'.

Comment	Name	Password	Service	Caller ID	Profile	Local Address	Remote Address	Routes	Last Logged Out
	Victor	*****	pptp		VPN-enc				1970-01-01 00:00:00

Lo siguiente que hacemos es ir a Interface y decirle que queremos que se active el servidor PPTP Server

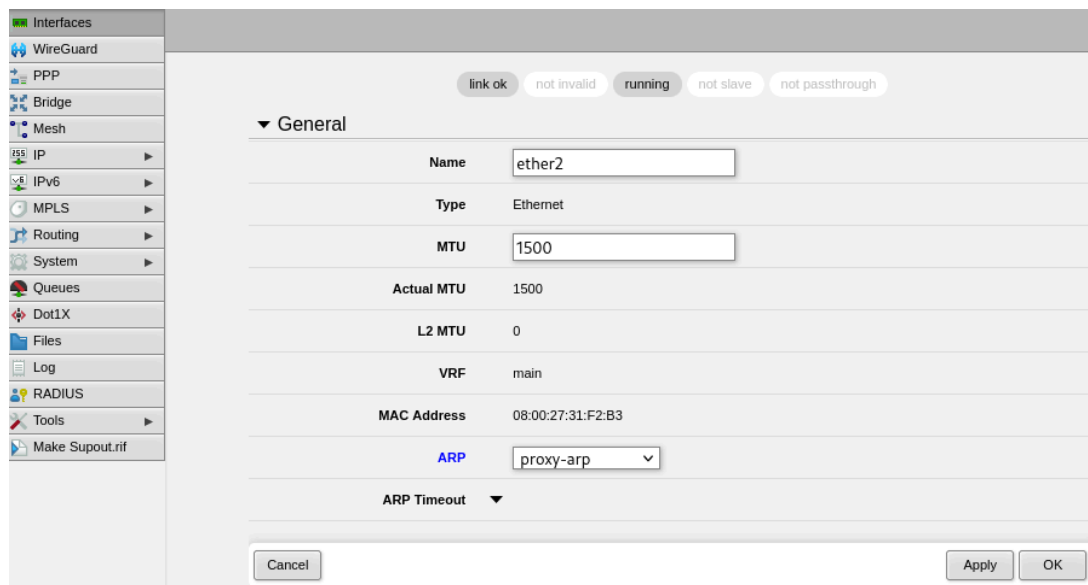


Pulsamos sobre este y marcamos el tick de Enabled y Default Profile seleccionamos VPN-enc



Pulsamos OK

Por último lo que nos queda es volver a Interfaces y en el adaptador de red interna (ether2) y decirle que el arp sea proxy-arp.



Pulsamos luego OK

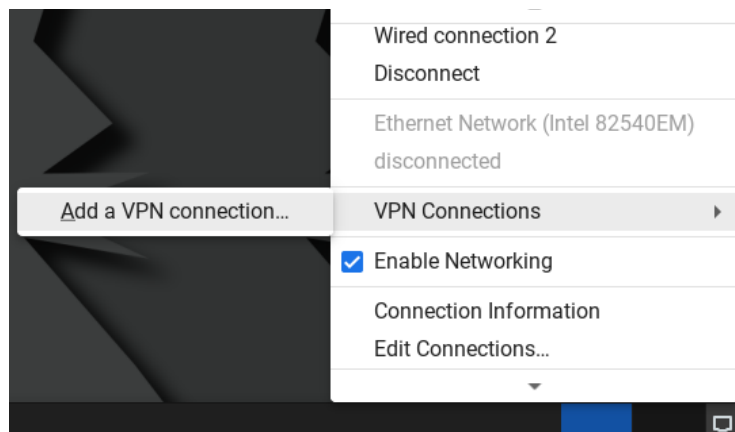
		Comment	Name	Type	Actual MTU	L2 MTU	Tx	Rx	Tx Packet (p/s)
D	R		ether1	Ethernet	1500		7.4 kbps	1056.6 kbps	14
D	R		ether2	Ethernet	1500		1059.7 kbps	7.9 kbps	88
-	D		ptp-in1	PPTP Server Binding			0 bps	0 bps	0

De esta manera ya tenemos configurado nuestro router MikroTik para que actúe como servidor VPN.

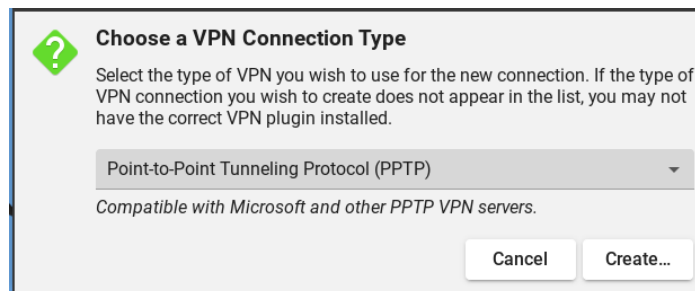
Ahora vamos a configurar una máquina cliente, que bien podría ser una máquina Linux o Windows.

8.4. Conexión a la VPN

Para esta prueba tenemos montado nuestro cliente con una tarjeta de red capaz de ver a nuestro mikrotik a través de otra tarjeta de red.



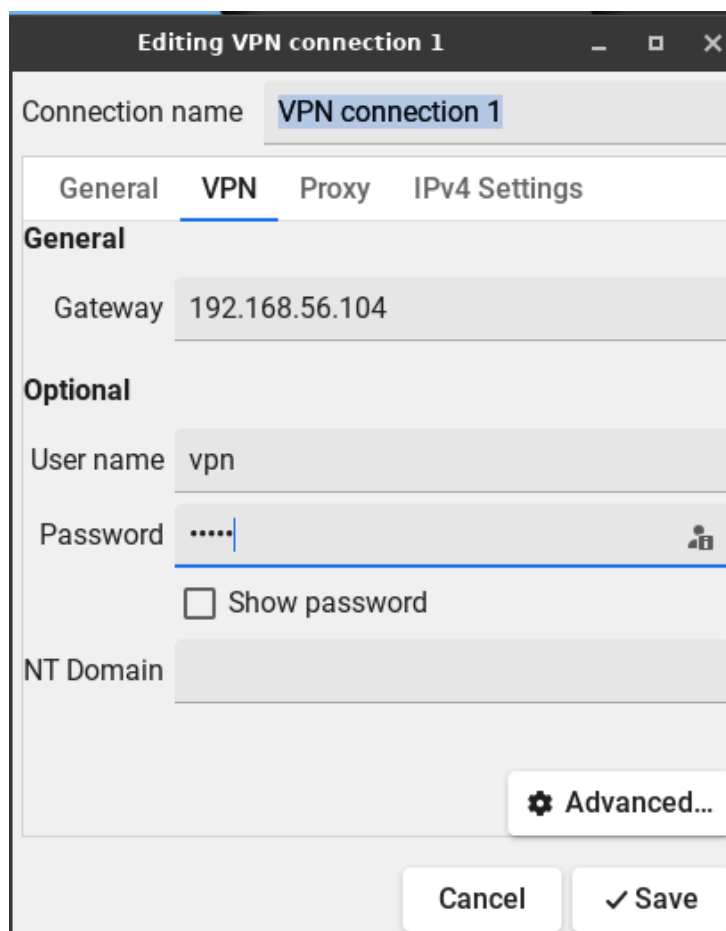
El tipo de conexión será de Protocolo de túnel punto a punto (PPTP)



Pulsamos a Crear ...

En el gateway colocamos la dirección ip del router MikroTik que está expuesta al exterior. En un entorno real sería la dirección IP del router Mikrotik que está visible en Internet.

Le colocamos el usuario que hemos creado, vpn, y la contraseña, marcando en el icono de la persona a la derecha de la casilla de introducción de la contraseña que es para ese usuario sólo, la contraseña, 12345.



Ahora, si nos conectamos podemos ver la ip que nos asigna del pool para VPN que va desde 192.168.100.220 a 192.168.56.211.

```
linuxlite ~$ ip a show ppp0
5: ppp0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1400 qdisc f
UNKNOWN group default qlen 3
    link/ppp
    inet 192.168.100.218 peer 192.168.100.210/32 scope global ppp0
        valid_lft forever preferred_lft forever
    inet6 fe80::806d:4589:f963:9c34 peer fe80::f0:2/128 scope link
        valid_lft forever preferred_lft forever
```

Como se ve, tenemos la nueva interfaz PPP0, lo que indica conexión a la VPN, y una dirección ip del rango .211 - 220 que indicamos en el pool del DHCP para estas conexiones.

9. Recursos

Para la realización del proyecto, he requerido de recursos físicos como mi ordenador portátil, para simular un entorno real ya que no cuento con los recursos para hacerlo real. Para ello he empleado máquinas virtuales para instalar los software necesarios.

9.1. Herramientas hardware

Portatil	ASUS Zenbook 14
Procesador	11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz
RAM	16,0 GB
SSD	512
Cámaras	Tapo C200

9.2. Herramientas software

Sistema operativo servidores	ubuntu 22.04.4 live server
Sistema operativo Router	MikroTik RouterOS CHR v7.11.2
Sistema operativo cliente	linux lite 6.6-64 bit
Sistema operativo dockers	ubuntu-22.04.4-LTS
Servicio DNS	Bind-9.18.18
Balanceador de carga	Keepalived-2.2.4
Base de datos	MySQL-8.0.36
Base de datos PHP	Redis-6.0.16
Servicio web	Apache-2.4.58
Version PHP	php8.1-fpm

10. Conclusiones

10.1. Conclusiones sobre el trabajo realizado

Durante el desarrollo del proyecto, inicialmente me enfoqué en la creación de la interfaz web. Sin embargo, conforme avanzaba, surgieron diversas consideraciones estratégicas. Una de ellas fue la decisión de realizar pruebas con cámaras IP públicas antes de implementar las cámaras reales. Al montar el balanceador de carga, enfrenté dificultades con la transferencia de sesiones PHP entre los servidores, las cuales se resolvieron gracias a la implementación de Redis para la gestión de sesiones. Además, la integración de las cámaras presentó desafíos adicionales.

10.2. Conclusiones personales

Este proyecto ha sido una experiencia de aprendizaje significativa para mí. He profundizado mis conocimientos en una variedad de tecnologías, explorando conceptos como la gestión de procesos en PHP, el protocolo de comunicación HTTP2, la base de datos en memoria Redis y la orquestación de contenedores con Docker Compose. Además, esta oportunidad me ha permitido adentrarme en la integración de cámaras IP y el desarrollo de interfaces de usuario dinámicas.

10.3. Posibles ampliaciones y mejoras

Para futuras mejoras, considero importante desarrollar una página específica para mejorar la gestión de los datos provenientes de las cámaras, así como implementar un firewall que bloquee todas las direcciones IP excepto las de la red local para fortalecer la seguridad. Además, se podría agregar un switch físico para facilitar la configuración de VLANs, lo que contribuiría a una administración más eficiente de la red.

11. *Anexos y documentos complementarios*

Junto a esta misma memoria, adjunto el enlace directo a un repositorio GitHub que contienen el siguiente contenido:

- Una copia en formato PDF de este documento
- Carpeta de imágenes y capturas utilizadas en esta memoria.
- Un respaldo del código fuente de la página web junto con la base de datos.
- Cada archivo de configuración mostrado en esta memoria

<https://github.com/victorbr04/proyecto.git>

12. *Webgrafía*

https://es.wikipedia.org/wiki/C%C3%A1mara_IP

<https://zoominformatica.com/blog/como-insertar-una-camara-ip-en-nuestra-pagina-web/>

<https://www.digitalocean.com/community/tutorials/how-to-set-up-a-redis-server-as-a-session-handler-for-php-on-ubuntu-14-04>

<https://www.wallarm.com/what/what-is-http-2-and-how-is-it-different-from-http-1>

<https://www.foscam-online.es/how-to/C%C3%B3mo-usar-RTSP-HTTPS-c%C3%A1maras-HD.pdf>

<https://www.mvteamcctv.com/es/news/What-s-the-RTSP-URL-of-IP-Camera-and-how-to-play-it-in-VLC-Media-Player.html>

<https://www.keepalived.org/index.html>