PLATAFORMA DE MONITORIZACIÓN Y ALERTAS PARA SERVIDORES DE EMPRESA

Mónica Prieto Cardeiro

2025/2026

1. Contextualización

1.1. Presentación del proyecto

El proyecto que se plantea consiste en la implantación de una plataforma de monitorización centralizada, está destinado a microempresas y PYMES que necesiten supervisar sus sistemas informáticos de una forma sencilla y eficaz.

Su propósito principal es garantizar la disponibilidad y seguridad de la infraestructura permitiendo detectar incidencias de forma rápida, generar alertas automáticas y aplicar medidas de seguridad y copias de respaldo que aseguren la continuidad del servicio.

Los requisitos básicos que deberá cumplir son:

- Ser accesible vía web para los usuarios autorizados.
- Monitorizar los recursos principales del servidor (CPU, memoria, disco, red) y los servicios desplegados en contenedores.
- Contar con un sistema de alertas configurables que notifique incidencias relevantes.
- Incluir medidas de seguridad y copias de respaldo.
- Ser escalable, permitiendo añadir más servidores o servicios en caso de crecimiento.

1.2. Contextualización

Para este trabajo tenemos como escenario una pequeña empresa que dispone en la actualidad de un servidor principal con diferentes servicios desplegados en contenedores y un NAS Synology, utilizado para almacenamiento y copias de seguridad. Situados en redes distintas y protegidos mediante un firewall Fortinet.

El modelo de negocio se centra en la optimización de procesos internos y asegurar que los sistemas informáticos sigan funcionando para que la empresa no tenga interrupciones en su trabajo diario.

A medio plazo, se espera que la infraestructura pueda ampliarse, añadiendo más servidores o abriendo servicios básicos para clientes, como páginas web o almacenamiento compartido.

La solución se desplegará en un nodo independiente, un equipo dentro de la red, que actuará como servidor de monitorización. En el servidor principal y en el NAS Synology se instalarán los exporters necesarios para enviar métricas. Esta arquitectura permite que, en caso de caída del servidor principal, el sistema de monitorización siga operativo y pueda generar alertas.

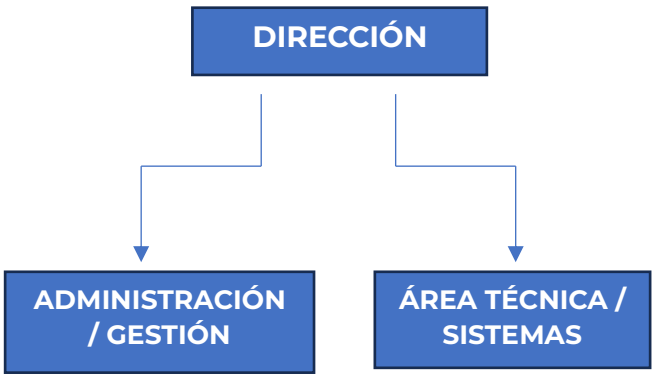
Además de la monitorización, contempla medidas de seguridad (bastionado y control de accesos), copias de respaldo y la posibilidad de gestión y acceso remoto seguro, reforzando la continuidad del servicio.

La plataforma está pensada para adaptarse a distintos tipos de organización.

Puede integrarse tanto en empresas del sector tecnológico como en compañías de otros ámbitos que necesiten supervisar sus sistemas de forma centralizada y segura.

La implantación estaría a cargo del área técnica o de sistemas.

A continuación, se muestra un organigrama genérico de implantación de la solución:



1.2.1. Descripción externa

La empresa no comercializa un producto al público general, sino que se centra en garantizar servicios internos que sean estables. El valor que aporta es la disponibilidad y seguridad de sus sistemas, lo que garantiza que la productividad de los empleados no se vea interrumpida por errores o caídas en los servicios.

1.2.2. Descripción interna

La empresa está formada por una única sede, con un equipo reducido de empleados. El área técnica está a cargo de un administrador de

sistemas, que es el que lleva el mantenimiento de la infraestructura. El resto del personal se dedica a funciones de gestión.

1.2.3. Viabilidad y expansión del negocio

La solución de monitorización es viable y escalable, ya que se apoya en herramientas libres y puede crecer en paralelo con la empresa.

- A corto plazo, permitirá controlar el servidor y sus servicios junto al NAS.
- A medio plazo, podrá incorporar otros servidores o servicios críticos.
- A largo plazo, la plataforma puede ampliarse para ofrecer servicios a clientes externos o integrarse con infraestructuras más grandes, sin necesidad de inversiones elevadas.

1.3. Descripción e objetivos del proyecto

La motivación de este proyecto surge de la falta de un sistema centralizado de monitorización en pequeñas empresas, que suelen depender de revisiones manuales o de que los usuarios reporten incidencias. Esto provoca retrasos en la detección de problemas, pérdidas de productividad y riesgo de indisponibilidad de servicios críticos.

En el mercado existen soluciones consolidadas como Zabbix o Nagios, que permiten cubrir estas necesidades, pero suelen estar orientadas a entornos más grandes y complejos, con despliegues pesados y funciones algo grandes para una pyme.

Por eso se propone una solución más ligera, modular y escalable adaptada al contexto de microempresas y pymes.

Estas medidas garantizan que la plataforma no solo detecte problemas, sino que también contribuya a mantener la continuidad del servicio y la resiliencia de la infraestructura

Esta solución es una posible oportunidad de negocio ya que se puede replicar en otras microempresas y pymes.

Se podría hacer una implantación on-premise o por suscripción (SaaS).

1.3.1. Necesidades que cubre el proyecto

1.3.2. Objetivo general

Implantar una plataforma de monitorización centralizada y segura que ayude a microempresas y pymes a controlar en tiempo real el estado de sus servidores y servicios. Además, la solución permitirá recibir alertas automáticas cuando haya incidencias, aplicar medidas básicas de seguridad y resiliencia, y contar con copias de seguridad y procedimientos de restauración que aseguren que la empresa pueda seguir trabajando sin interrupciones.

1.3.3. Objetivos específicos

- Monitorizar los recursos básicos del servidor (CPU, memoria, disco, red).
- Monitorizar servicios desplegados en contenedores y el NAS de almacenamiento.
- Configurar un sistema de alertas que notifique incidencias críticas (caída de servicio, uso excesivo de CPU, espacio en disco insuficiente, latencia elevada).
- Control de acceso seguro al servidor y a los paneles (SSH por clave más credenciales protegidas en Grafana.
- Aplicar medidas de bastionado: firewall, cierre de puertos no utilizados, contraseñas robustas y claves seguras.
- Implementar un sistema de copias de seguridad y realizar al menos una prueba de restauración.
- Documentar la instalación, la operación de la plataforma y los procedimientos de respuesta ante incidentes.
- *(Opcional)* Evaluar el uso de una herramienta ligera de monitorización en tiempo real (Netdata) como complemento de la solución principal.

La solución se basará en herramientas de software libre utilizadas habitualmente en la administración de sistemas y en la monitorización de infraestructuras TI. Se empleará un sistema operativo Linux como base para el despliegue, junto con servicios de contenedorización (por ejemplo, Docker y Docker Compose) que permitan un entorno reproducible y seguro.

Para la parte de monitorización y alertas se utilizarán plataformas de recogida y visualización de métricas (como Prometheus, Grafana o soluciones equivalentes), así como sistemas de notificación de incidencias. Se incorporarán también medidas de

seguridad y bastionado (firewall, políticas de contraseñas y copias de seguridad) que garanticen la continuidad de los servicios.

2. Análisis

2.1. Requisitos de la solución técnica

A continuación, se describen los requisitos técnicos y operativos que debe cumplir la plataforma de monitorización.

- **Plataforma de monitorización centralizada**

La solución deberá permitir monitorizar en tiempo real los servidores y servicios de la empresa, integrando en un mismo panel las métricas de rendimiento, consumo de recursos y estado de los servicios críticos.

- **Sistema de alertas y notificaciones**

La plataforma deberá generar alertas automáticas cuando se produzcan incidencias relevantes como pueden ser: caída de un servicio, uso excesivo de CPU, falta de espacio en disco, latencia elevada, se notifica al administrador mediante diferentes canales configurables.

- **Seguridad y control de acceso**

Se implementarán medidas de bastionado y control de acceso. El acceso al servidor y a los paneles de monitorización se realizará mediante autenticación segura (clave SSH), restringiendo los permisos según el rol de cada usuario.

- **Copias de respaldo y recuperación**

La solución deberá incluir políticas de copia de seguridad, almacenando las copias en el NAS Synology y verificando su correcta restauración.

- **Despliegue en contenedores**

El sistema se ejecutará sobre contenedores Docker, utilizando Docker Compose para simplificar la instalación, mantenimiento y reproducibilidad del entorno.

▪ **Herramientas**

Se emplearán herramientas libres como Prometheus, Grafana, Alertmanager por su compatibilidad con entornos de contenedores y su carácter open source, así se evitan costes de licencia y facilita la escalabilidad en el futuro.

▪ **Infraestructura y red**

La implantación se realizará en entorno on-premise, en un nodo independiente dentro de la red local de la empresa, conectado al servidor principal y al NAS Synology.
El NAS será monitorizado mediante protocolo SNMP, y los contenedores se comunicarán entre sí a través de una red privada definida en Docker.

▪ **Autenticación y usuarios**

Los usuarios tendrán diferentes perfiles de acceso. La autenticación será local, aunque se prevé la posibilidad de integración futura con un servicio de directorio (LDAP o Active Directory).

▪ **Escalabilidad y resiliencia**

El sistema deberá poder adaptarse fácilmente al crecimiento de la infraestructura, en caso de que se añadan nuevos servidores o servicios.

▪ **Políticas de software y licencias**

Todas las herramientas utilizadas serán de código abierto, lo que garantiza libertad de uso, modificación y distribución, reduciendo costes y favoreciendo la sostenibilidad del proyecto.

3. Planificación temporal y de recursos

3.1. Identificación de las actividades

Se detallan a continuación las principales actividades del proyecto

Fase 1 – Preparación

Revisión y análisis del entorno existente (servidor Proxmox, NAS, Fortinet) Definición de las necesidades de monitorización y de la arquitectura del miniservidor.

Creación de la estructura base del entorno y planificación del direccionamiento.

Duración estimada -> 2 jornadas

Fase 2 – Implementación del entorno base

Configuración inicial de la red del miniservidor.
Instalación del sistema operativo Ubuntu Server 22.04.
Instalación de Docker, Docker Compose y herramientas base.
Configuración del acceso seguro mediante SSH con clave pública.
Aplicación del bastionado básico inicial.

Duración estimada -> 1 jornada

Fase 3 – Seguridad

Refuerzo de la seguridad del miniservidor:

- usuario técnico restringido
- servicio SSH endurecido
- permisos mínimos
- separación de redes y acceso controlado por Fortinet

Duración estimada -> media jornada

Fase 4 – Integración y configuración de los servicios de monitorización

Instalación y despliegue en Docker de:

- Prometheus
- Alertmanager
- Grafana

Integración de métricas mediante exporters:

- Node Exporter (miniservidor y Proxmox)
- SNMP Exporter (NAS Synology con SNMPv3)
- Blackbox Exporter (verificación de accesibilidad)

Configuración de reglas de alertas (CPU, RAM, disco, servicios). Creación de dashboards personalizados en Grafana para Proxmox, NAS y LXC.

Duración estimada -> dos jornadas

Fase 5 – Pruebas y documentación

Pruebas de funcionamiento de métricas y exporters.
Simulación de alertas y verificación de notificaciones.
Comprobación del almacenamiento de backups y persistencia de volúmenes.
Redacción de documentación final del proyecto.

Duración estimada -> 3 jornadas

Duración total estimada: 16 jornadas (128 horas de trabajo).

3.2. Identificación de recursos

Recursos de personal

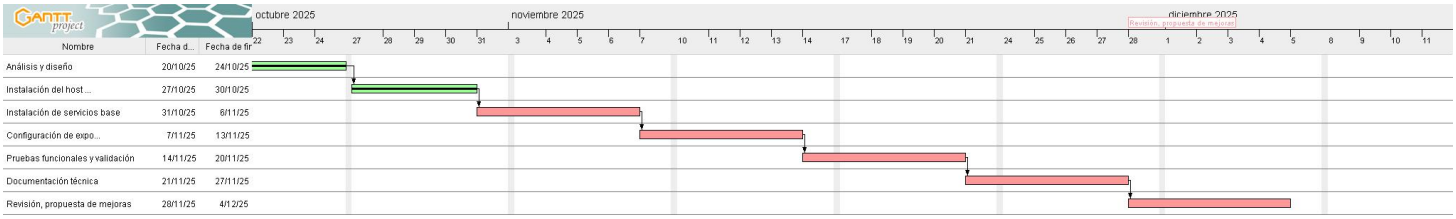
ROL	TAREAS PRINCIPALES	CUALIFICACIÓN
Administrador de sistemas	Análisis, diseño, instalación, configuración, pruebas y documentación.	Técnico Superior en Administración de Sistemas Informáticos en Red

Recursos materiales

TIPO	DESCRIPCIÓN	OBSERVACIONES
Servidor Principal (PROXMOX)	Host físico donde están los contenedores monitorizados.	CPU 8 núcleos, RAM 16 GB, almacenamiento 1 TB SSD.
NAS Synology	Almacenamiento de copias de seguridad y métricas.	IP 172.16.4.2 (subred separada, gestionada por Fortinet).
Equipo de monitorización	PC o mini servidor con Prometheus, Grafana, Alertmanager.	IP *192.168.20.21*, conectado la red LAN creada monitorización.
Firewall Fortinet	Segmentación de redes y control de acceso.	Permite comunicación segura entre subredes.
Software	Prometheus, Grafana, Alertmanager, Docker, Docker Compose, Debian/Ubuntu.	Todo software libre y multiplataforma.

3.3. Planificación temporal

A continuación, se muestra el diagrama de Gantt que refleja la planificación del proyecto.



3.4. Presupuesto

Este proyecto se ha diseñado para ser implementado íntegramente con software libre y aprovechando infraestructura existente, esto hace que los costes sean más bajos.

Se incluye también una estimación del coste en un escenario real de empresa en el que fuera necesario adquirir por ejemplo un miniservidor para la monitorización.

Coste de recursos humanos

El proyecto ha sido realizado por una única persona con perfil de administradora de sistemas, estimando una dedicación total de 128 horas de trabajo (16 jornadas).
Se aplica una tarifa profesional media de 20 €/hora, lo que equivale a:

128 h × 20 €/h = 2.560 €

Coste de recursos materiales

ELEMENTO	DESCRIPCIÓN	COSTE	OBSEVACIONES
Servidor principal (Proxmox)	Ya existe en la empresa	0€	No se requiere inversión
NAS Synology	Almacenamiento de copias de seguridad		Equipo ya disponible

Equipo de monitorización	Mini servidor	500€	
Firewall Fortinet	Equipo ya disponible.		Equipo ya disponible
Software	Prometheus, Grafana, Alertmanager, Docker, Debian, SNMP, etc		Todo software libre
Electricidad y mantenimiento	Consumo estimado durante las pruebas y funcionamiento	40€	
Total recursos :		**540€**	

Recursos humanos -> 2560€

Recursos materiales -> 540 €

Total -> 3100€

4. Diseño de la solución

4.1. Diseño general del sistema

En el siguiente esquema se muestra cómo se conectan los distintos elementos: el servidor principal con sus contenedores, el NAS Synology donde se almacenan las copias, el firewall Fortinet que protege las redes y el equipo de monitorización desde el que se controlan todos los servicios.

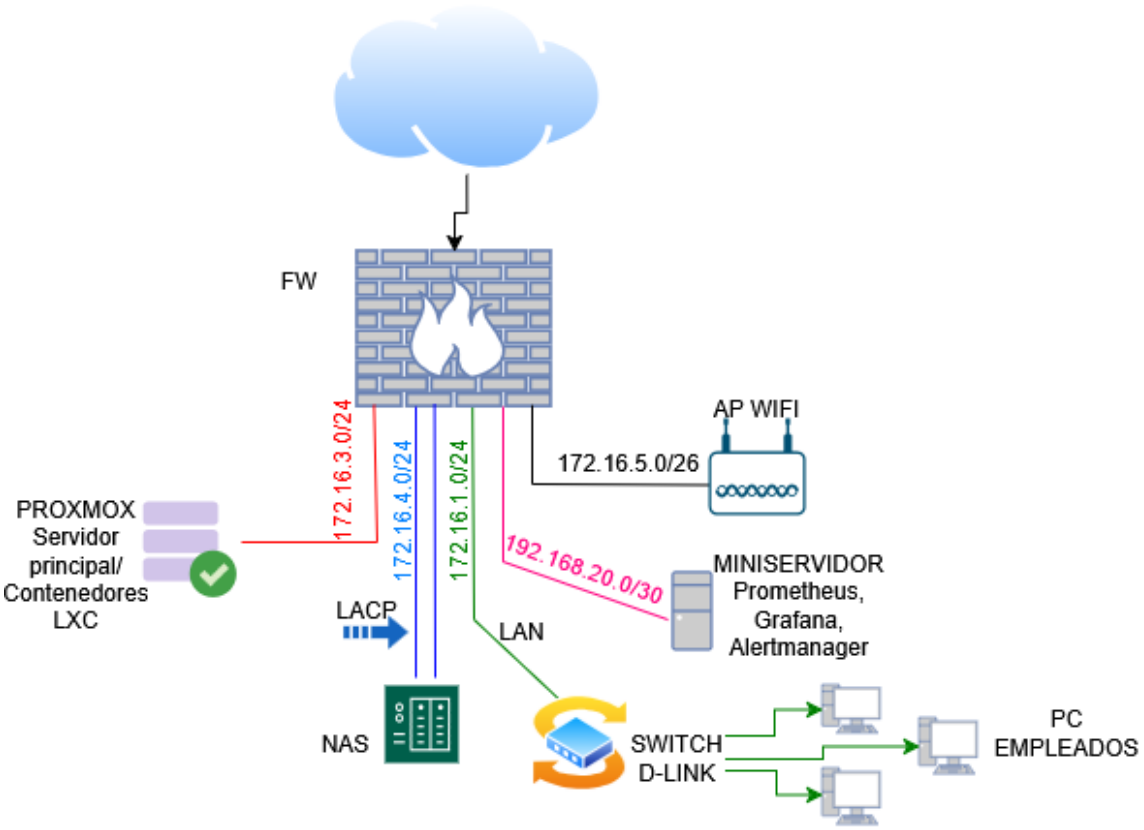


Figura 1. Arquitectura general del sistema

En la Figura 1 se muestra la arquitectura general del sistema. Vemos la segmentación de la red en cuatro subredes principales:

- **172.16.3.0/24**: red para los servidores, en este caso solo se dispone de uno, Proxmox, con los contenedores LXC que ejecutan los distintos servicios corporativos.
- **172.16.4.0/24**: red de almacenamiento, la utiliza el NAS Synology para las copias de seguridad y el almacenamiento de métricas.
- **172.16.5.0/26**: red WIFI.
- **192.168.20.0/26:** red de monitorización, dedicada al miniservidor donde se ejecutan Prometheus, Grafana y Alertmanager, garantizando una conexión estable y aislada del resto de redes.
- **172.16.1.0/24**: red corporativa, conecta los equipos de los empleados, impresoras y otros dispositivos de trabajo. El firewall controla el acceso de esta red hacia los servicios internos.

El firewall Fortinet actúa como punto de control del tráfico entre redes y acceso a Internet, garantizando la seguridad y el aislamiento de los distintos segmentos.

El sistema permite que el equipo de monitorización recopile métricas del servidor principal y del NAS incluso si alguno de los servicios deja de estar disponible, asegurando la continuidad de la supervisión y la generación de alertas.

4.2. Diseño del miniservidor

Este apartado describe la arquitectura interna del miniservidor de monitorización.

El miniservidor actúa como nodo central de monitorización dentro de la infraestructura.
En él se despliegan los servicios principales mediante contenedores Docker, gestionados con Docker Compose, lo que permite aislar cada componente, facilitar actualizaciones y mejorar la portabilidad de la solución.

Sistema base

- Sistema operativo: Linux (Ubuntu/Debian).
- Usuario dedicado para administración y conexión segura por SSH.
- Estructura de directorios:

[admin@monitoring:~/monitoring]$ tree

.

├── alertmanager

│ └── alertmanager.yml

├── blackbox

│ └── config.yml

├── docker-compose.yml

├── exporters

├── grafana

├── prometheus

│ ├── alerts.yml

│ └── prometheus.yml

└── snmp_exporter

└── snmp.yml

Estructura de red interna:

Todos los contenedores comparten una misma red Docker tipo *bridge*, denominada monitoring, que permite la comunicación segura entre los servicios. Solo se exponen a la LAN los puertos estrictamente necesarios. (Prometheus, Grafana, Alertmanager y exporters).

Además, el miniservidor está conectado a la red local de la empresa (LAN) a través de la interfaz lan dedicada a monitorización, que proporciona conectividad tanto hacia los equipos internos como hacia Internet.

Prometheus -> puerto 9090

Grafana -> puerto 3000

Alertamanager -> 9093

Exporters -> node/snmp

5. Manual de implantación

En la planificación (punto 3.1), la Fase 1 se denomina *"Preparación"* e incluye la revisión y análisis del entorno.

En esta guía de implantación, esa fase se desarrolla de manera práctica bajo el título *"Preparación del entorno"*, que abarca la instalación, configuración inicial y bastionado del sistema.

Se mantiene la correspondencia entre las fases descritas en ambos apartados, aunque los nombres puedan variar ligeramente por motivos de claridad técnica.

Antes de proceder con la implementación práctica, se realizó una revisión y análisis del entorno existente para determinar las necesidades de la empresa en materia de monitorización y seguridad.

De esta evaluación se concluyó que:

- No existía un sistema centralizado de monitorización ni de alertas.
- El servidor principal y el NAS funcionaban correctamente, pero sin control automatizado de recursos.
- Era necesario implantar un sistema que detectara incidencias de forma proactiva y notificara automáticamente a los técnicos.
- Se requería una solución ligera, portable y fácilmente ampliable, compatible con la infraestructura actual basada en contenedores.

Estas conclusiones sirvieron de base para definir los requisitos del proyecto y planificar la preparación del entorno que se desarrolla en los apartados siguientes.

5.1. Preparación del entorno

Antes de empezar con la instalación de los servicios de monitorización, se prepara el entorno de trabajo en el miniservidor.

En esta parte se configuraron las redes necesarias, se activó el acceso seguro por SSH y se instalaron las herramientas básicas que servirán para montar Prometheus, Grafana y Alertmanager dentro de contenedores Docker.

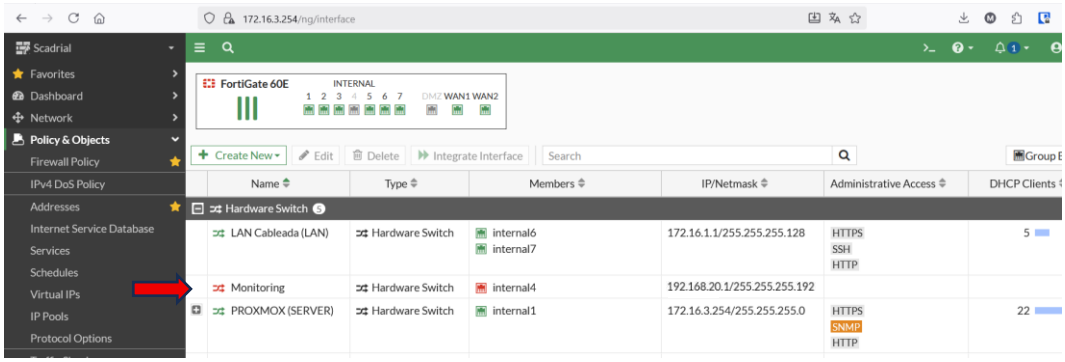
Se creó una estructura de directorios en el sistema para organizar todos los archivos del proyecto.

5.1.1. Configuración de red y direccionamiento

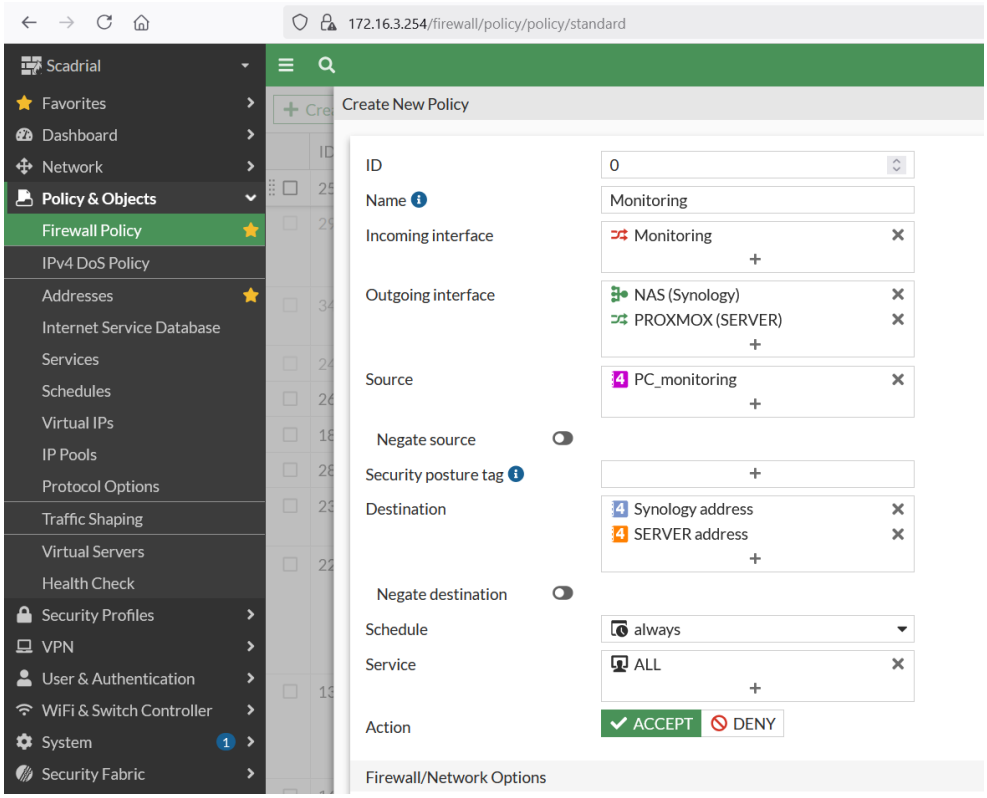
Se creó en el firewall Fortinet una nueva red dedicada (192.168.20.0/26) destinada al miniservidor de monitorización, asegurando una conexión cableada e independiente.

Al miniservidor se le asignó la IP fija 192.168.20.21, con acceso controlado únicamente a las redes de servidores (172.16.3.0/24) y almacenamiento (172.16.4.0/24)

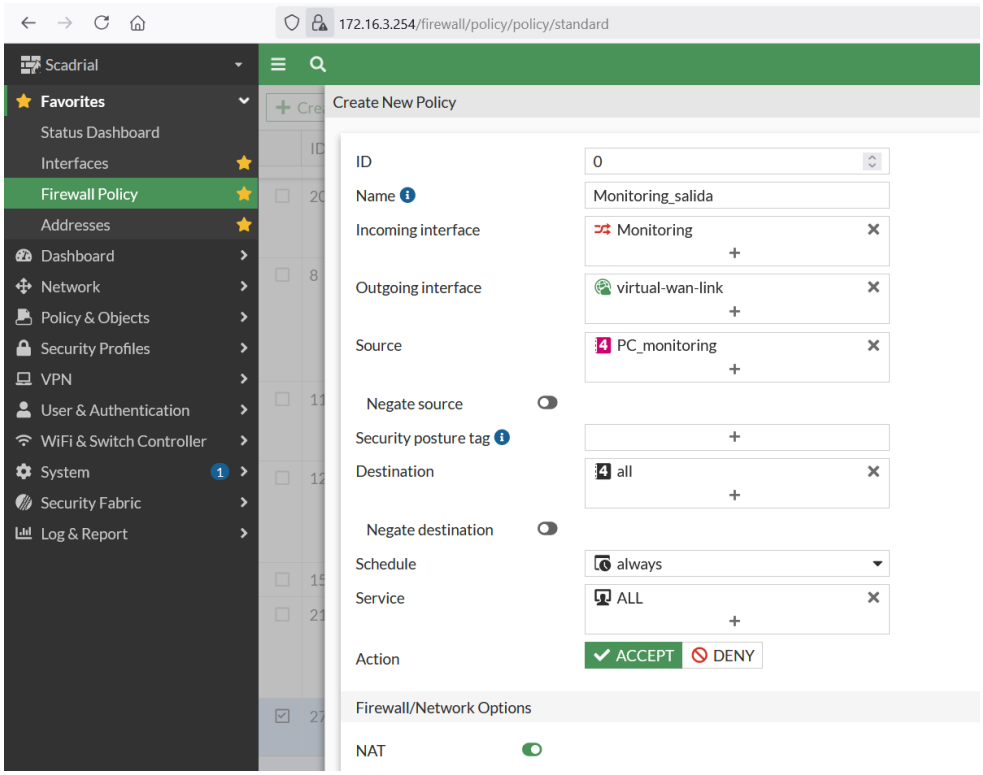
Se configura la red 192.168.20.0/26

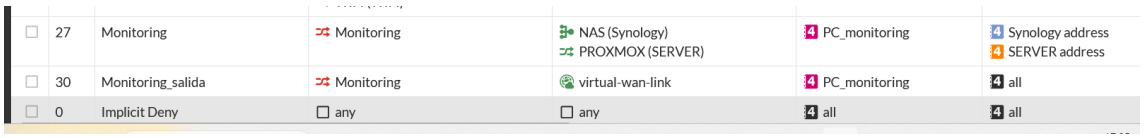


Posteriormente creamos una política de acceso en el Fortinet para permitir visibilidad entre las redes internas, de forma que el miniservidor pueda acceder al servidor Proxmox y al NAS.



Creamos también una regla específica que permite el acceso a Internet desde el equipo de monitorización.





5.1.2. Instalación de Docker y dependencias

Instalación del sistema operativo.

Para el desarrollo y despliegue de la plataforma se utilizó un sistema operativo Ubuntu Server 22.04 LTS, por su estabilidad, amplia compatibilidad con Docker y soporte a largo plazo.

Durante la instalación se realizaron las siguientes configuraciones básicas:

- Asignación de un nombre de host descriptivo para el equipo: monitoring.
- Configuración de la red con dirección IP estática dentro de la LAN local, garantizando conectividad con el resto de dispositivos y el NAS.
- Creación de un usuario técnico con privilegios administrativos (sudo) para la gestión de servicios: admin

Actualización sistema.

Antes de instalar Docker, se actualiza el sistema y se instalan las utilidades necesarias para asegurar las descargas y la verificación de paquetes:

sudo apt update && sudo apt upgrade

Para garantizar la seguridad de las conexiones HTTPS y las descargas desde repositorios externos, se instalaron los certificados actualizados del sistema mediante:

sudo apt install ca-certificates curl gnupg

Ca-certificates -> garantizan que las conexiones HTTPS sean seguras.

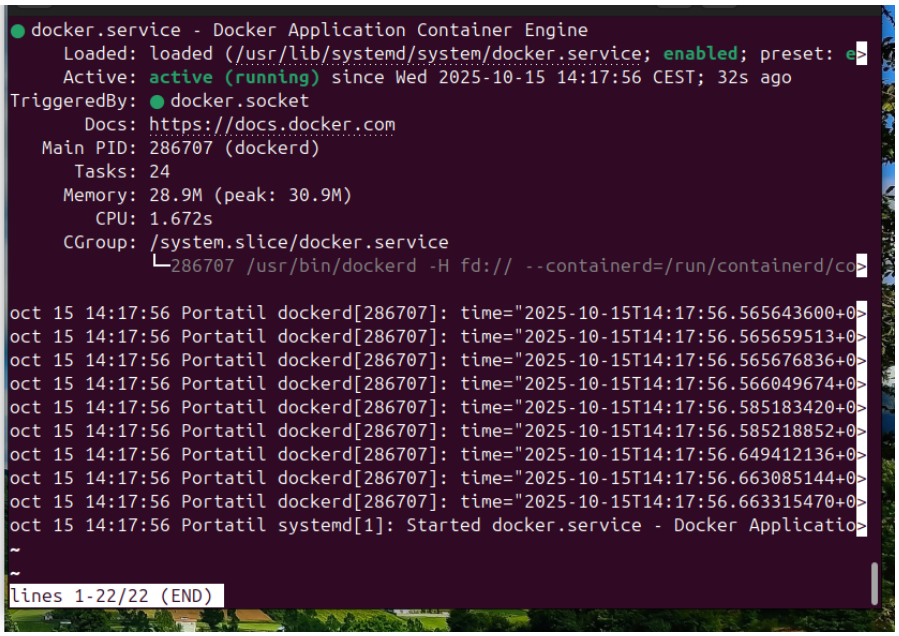
Curl -> permiten la descarga de archivos

Gnupg -> gestionan las claves de firma digital para validar la autenticidad del repositorio oficial de Docker.

Instalación Docker y Sistemas.

Se añadió el repositorio oficial y se instaló Docker y Docker Compose

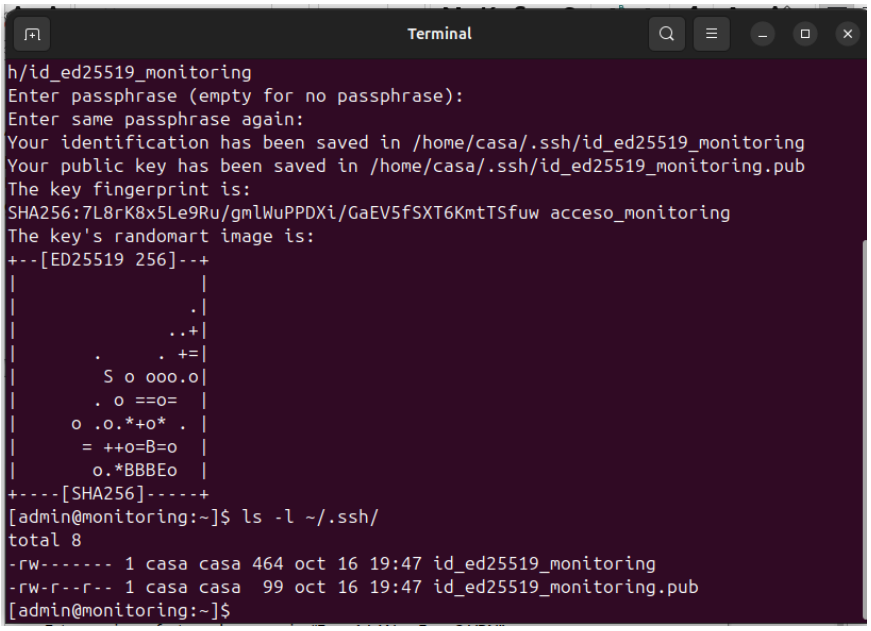
Comprobación del sistema.



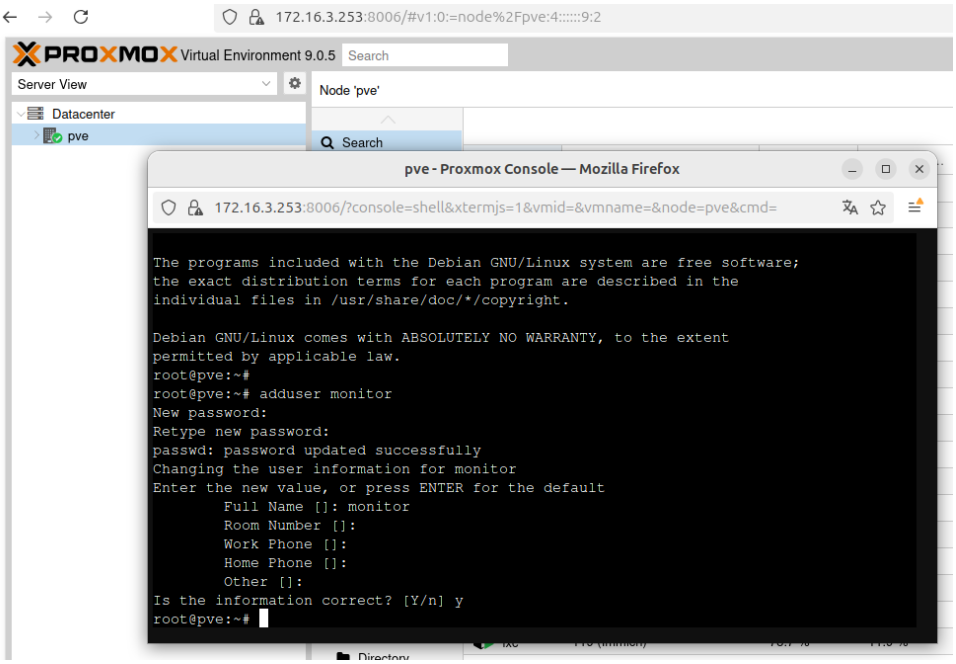
5.1.3. Seguridad básica

Generamos clave SSH en el pc desde el cual vamos a monitorizar y conectarnos al servidor. Se optó por el uso de autenticación SSH mediante clave pública en lugar de contraseña, debido a su mayor seguridad y comodidad en la automatización**.**

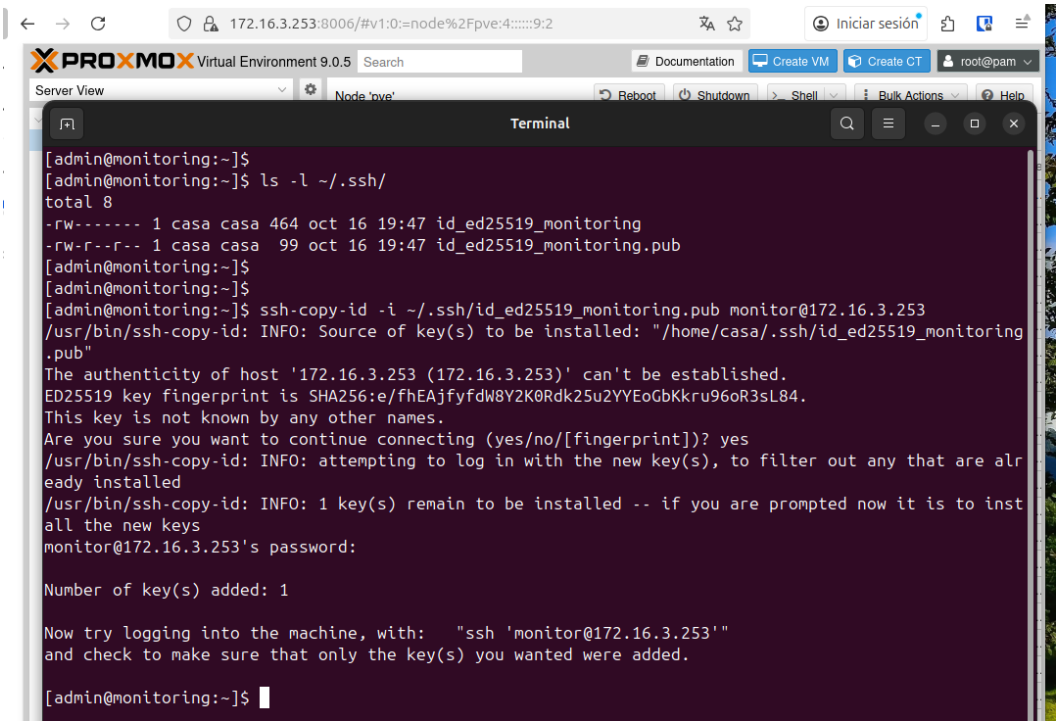
Este método elimina la necesidad de introducir contraseñas manualmente en cada conexión, evita ataques de fuerza bruta y permite una integración directa con las herramientas de monitorización



Tras generar las claves en el pc, procedemos a crear un usuario en proxmox:



Copiamos desde nuestro pc la clave ssh al servidor



Ajustamos permisos en el servidor

root@pve:~# chmod 700 ~/.ssh

root@pve:~# chmod 600 ~/.ssh/authorized_keys

Dentro del archivo de configuración del servidor: nano /etc/ssh/sshd_config

modificamos los siguientes parámetros para mayor seguridad.

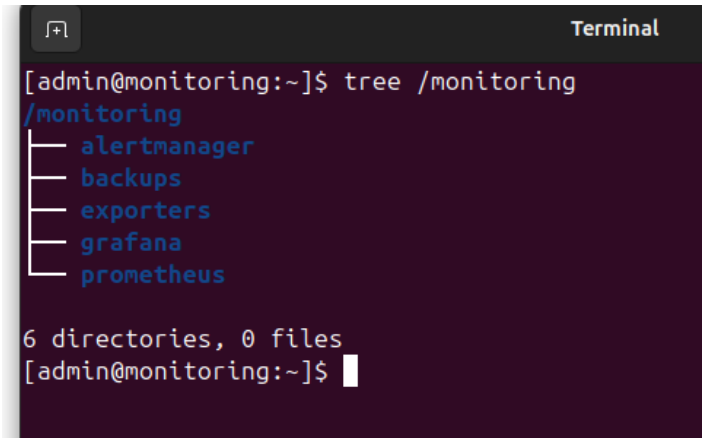
```
PubkeyAuthentication yes

PasswordAuthentication no
PermitRootLogin no
```

Tras aplicar los cambios, se reinició el servicio SSH con:

sudo systemctl restart ssh

5.2. Despliegue de los servicios de monitorización

Antes del despliegue de los servicios, se creó en el miniservidor la estructura de directorios donde se almacenarán las configuraciones, volúmenes y archivos necesarios para la plataforma de monitorización. La ruta utilizada fue ~/monitoring, añadiéndose una carpeta por cada componente principal:



Esta organización permite mantener separadas las configuraciones, facilita las copias de seguridad y mejora la claridad del despliegue.

Antes de aplicar la configuración completa del sistema de monitorización, se realizó un despliegue inicial con ajustes mínimos.

Esta primera fase tiene como objetivo:

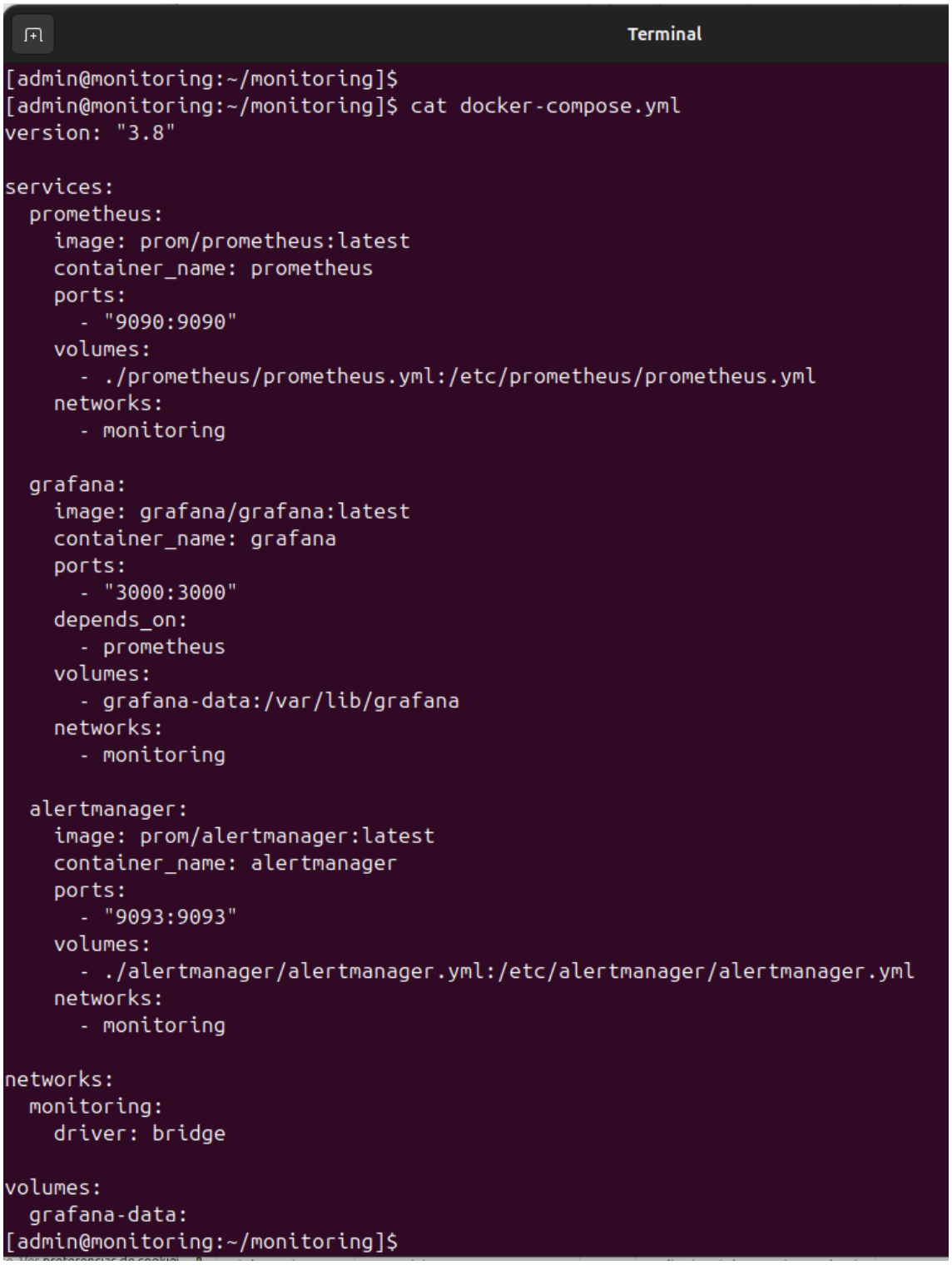
- validar el funcionamiento de los contenedores,
- comprobar la comunicación interna entre servicios (Prometheus, Grafana y Alertmanager),
- y asegurar que el entorno Docker está correctamente preparado antes de integrar métricas reales del servidor principal y del NAS.

Por este motivo, los archivos que se muestran a continuación (docker-compose.yml, prometheus.yml, alertmanager.yml) contienen una configuración simple.

Más adelante, estos archivos se ampliarán para incluir:

- los exporters reales del servidor principal y del NAS,
- reglas de alertas completas,
- dashboards personalizados en Grafana,
- y cualquier ajuste adicional necesario para la monitorización avanzada.

Para orquestar los tres servicios principales se utilizó Docker Compose. El archivo docker-compose.yml define la red interna, los contenedores y la persistencia de datos. Se crea archivo:



La red interna monitoring permite la comunicación segura entre los contenedores sin exponer servicios adicionales al exterior. Solo se publican los puertos necesarios para el acceso desde la LAN.

Configuración inicial de Prometheus

Se creó el archivo prometheus.yml con una configuración mínima que permite iniciar el servicio monitorizando su propio contenedor:

aquí captura de prometheus:

```
[admin@monitoring:~/monitoring]$
[admin@monitoring:~/monitoring]$ cd prometheus/
[admin@monitoring:~/monitoring/prometheus]$ cat prometheus.yml
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: "prometheus"
    static_configs:
      - targets: ["prometheus:9090"]
[admin@monitoring:~/monitoring/prometheus]$
```

Configuración inicial de Alertmanager

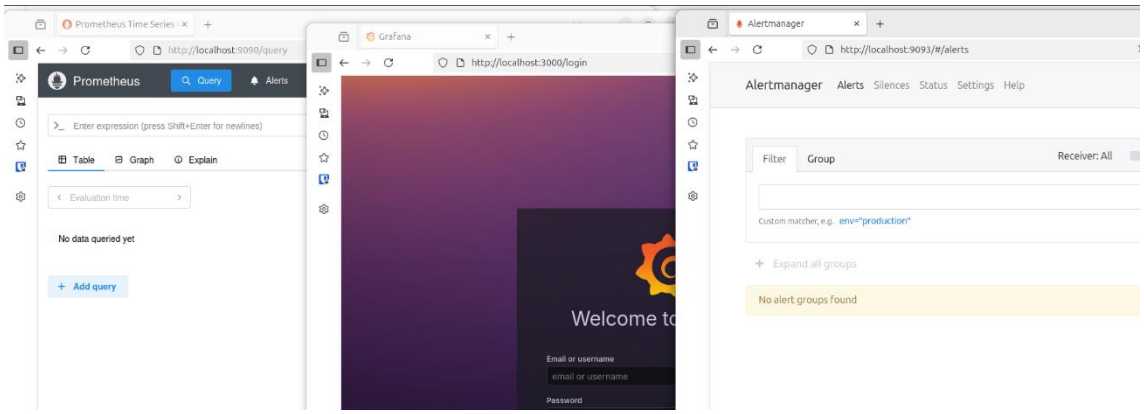
```
[admin@monitoring:~/monitoring/alertmanager]$
[admin@monitoring:~/monitoring/alertmanager]$ cat alertmanager.yml
global:
  resolve_timeout: 5m

route:
  receiver: "default"

receivers:
  - name: "default"
[admin@monitoring:~/monitoring/alertmanager]$
```

Una vez preparados los archivos, se desplegaron los contenedores con: docker compose up -d

Y tras el inicio, se comprobó el estado con: docker ps

```
Terminal
[admin@monitoring:~/monitoring]$
[admin@monitoring:~/monitoring]$ docker compose up -d
WARN[0000] /home/casa/monitoring/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 5/5
 ✔ Network monitoring_monitoring   Created                                0.2s
 ✔ Volume monitoring_grafana-data  Created                                0.0s
 ✔ Container prometheus            Started                                0.2s
 ✔ Container alertmanager          Started                                0.2s
 ✔ Container grafana               Started                                0.4s
[admin@monitoring:~/monitoring]$
[admin@monitoring:~/monitoring]$
[admin@monitoring:~/monitoring]$ docker ps
CONTAINER ID   IMAGE                     COMMAND                  CREATED          STATUS          PORTS                                                NAMES
52192f313da0   grafana/grafana:latest    "/run.sh"                14 seconds ago   Up 13 seconds   0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp          grafana
18313d971aa4   prom/prometheus:latest    "/bin/prometheus --c…"   14 seconds ago   Up 14 seconds   0.0.0.0:9090->9090/tcp, [::]:9090->9090/tcp          prometheus
d59bcb3c67e4   prom/alertmanager:latest  "/bin/alertmanager -…"   14 seconds ago   Up 14 seconds   0.0.0.0:9093->9093/tcp, [::]:9093->9093/tcp          alertmanager
[admin@monitoring:~/monitoring]$
[admin@monitoring:~/monitoring]$
```

Comprobamos que los servicios están operativos:



Grafana utiliza un volumen gestionado por Docker (grafana-data) para evitar problemas de permisos y asegurar que los dashboards y configuraciones persisten incluso tras reiniciar los contenedores.

Con este despliegue inicial podemos ver que la plataforma está correctamente instalada y preparada para recibir las métricas de los equipos reales.

5.2.1. Instalación Node_Exporter en el propio servidor.

Primero vamos a poner bajo vigilancia el propio miniservidor, con Node Exporter para comprobar funcionamieto.

Para permitir que Prometheus recopile métricas del propio miniservidor, se instaló el agente Node Exporter. Lo descargamos desde el repositorio oficial de Prometheus en GitHub y se despliega como un servicio del sistema para garantizar su ejecución automática y estable.

Ejecutamos en el miniservidor:

curl -LO https://github.com/prometheus/node_exporter/releases/download/v1.10.2/node_exporter-1.10.2.linux-amd64.tar.gz

y extraemos con:

tar xvf node_exporter-1.10.2.linux-amd64.tar.gz

Una vez extraído el binario lo movemos a /bin:

sudo mv node_exporter-1.10.2.linux-amd64/node_exporter /usr/local/bin/

Se movió el programa a una ruta estándar del sistema porque así cualquier servicio systemd o usuario puede ejecutarlo desde la ruta correcta.

CREACIÓN USUARIO SISTEMA.

Se creó un usuario del sistema por seguridad ya que Node_Exporter solo necesita leer información, no modificar nada.

Lo creamos con el comando:

sudo useradd -rs /bin/false node_exporter

CREACIÓN SERVICIO SYSTEMD.

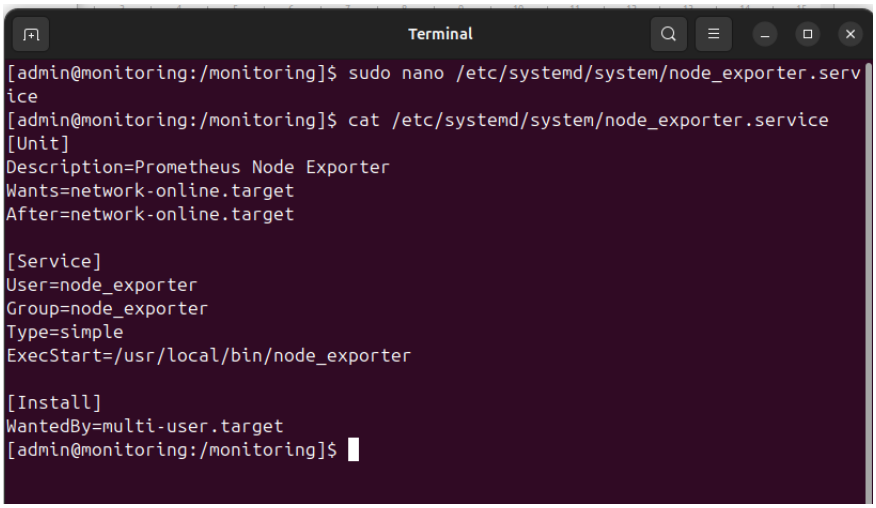
Systemd es el sistema que controla los servicios en Linux (como “Servicios” en Windows).

Con esto conseguimos que Node_Exporter arranque al iniciar el sistema, se ejecute en segundo plano y use el usuario restringido. Así Node Exporter es un servicio estable, no un simple archivo.

Se podrá controlar como un servicio norma con systemctl.

Creamos archivo con:

sudo nano /etc/systemd/system/node_exporter.service



se recarga el servicio:

sudo systemctl daemon-reload

sudo systemctl enable --now node_exporter

AÑADIR MINISERVIDOR COMO TARGET EN PROMETHEUS.

Editamos el fichero prometheus.yml creado anteriormente y añadimos el siguiente trabajo:

```
global:
  scrape_interval: 15s


scrape_configs:
  - job_name: "prometheus"
    static_configs:
      - targets: ["prometheus:9090"]


  - job_name: "miniservidor"
    static_configs:
      - targets: ["192.168.20.21:9100"]
```

Reiniciamos Prometheus: docker compose restart prometheus

Y al revisar la interfaz de prometheus (targets) vemos que está up.

5.2.2. Instalación Node_Exporter en el PROXMOX.

Los pasos son los mismos que en el apartado anterior pero ahora lo instalaremos en el proxmox.

ABRIR PUERTO 9100 EN EL FIREWALL.

Antes de proceder con la instalación de Node_Exporter, es necesario la apertura del puerto 9100.

Para que Prometheus pueda recopilar métricas del servidor Proxmox, es necesario que el miniservidor tenga acceso al puerto 9100/TCP, que es donde Node Exporter expone las métricas del sistema.

Como ambos equipos se encuentran en redes distintas y el tráfico debe atravesar el firewall Fortigate, fue necesario crear una política específica que permita ese flujo.

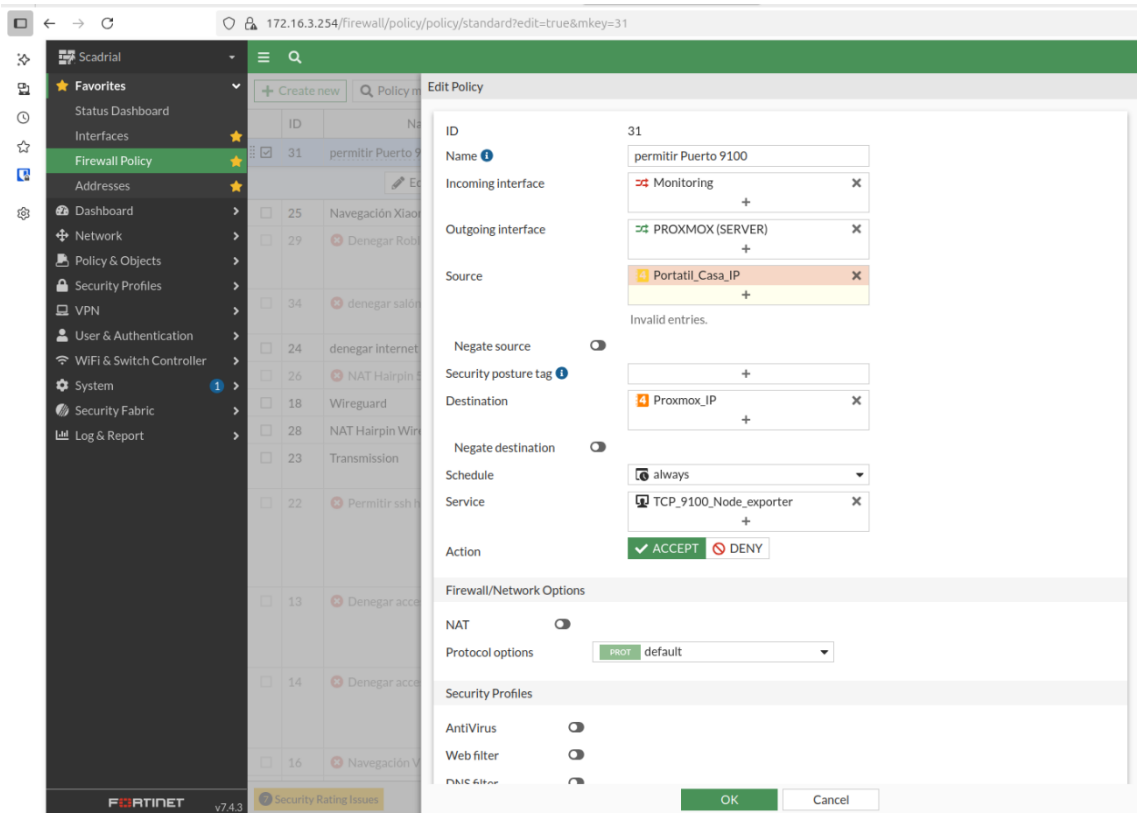
En el fortigate no existía ningún servicio

predefinido para el puerto 9100/TCP, por lo que fue necesario crear manualmente un nuevo servicio, definiendo:

- Protocolo: TCP
- Puerto de destino: 9100
- Nombre: TCP_9100_NodeExporter
-

Una vez creado este servicio, se añadió a la nueva regla que:

- permite conexiones desde la red del miniservidor hacia la red donde está Proxmox
- Solo para el servicio TCP/9100
- No se exponen otros puertos.



Una vez creada la subimos para que no quede en el fondo y le afecten reglas más restrictivas. Ahora procedemos con la instalación de node_exporter.

Se accede al proxmox y en consola descargamos:

cd /tmp

curl -LO https://github.com/prometheus/node_exporter/releases/download/v1.10.2/node_exporter-1.10.2.linux-amd64.tar.gz

Procedemos a descomprimir:

tar xvf node_exporter-1.10.2.linux-amd64.tar.gz

Una vez extraído el binario lo movemos a /bin:

mv node_exporter-1.10.2.linux-amd64/node_exporter /usr/local/bin/

CREACIÓN USUARIO SISTEMA.

useradd -rs /bin/false node_exporter

CREACIÓN SERVICIO SYSTEMD.

nano /etc/systemd/system/node_exporter.service

[Unit]

Description=Prometheus Node Exporter

Wants=network-online.target

After=network-online.target

[Service]

User=node_exporter

Group=node_exporter

Type=simple

ExecStart=/usr/local/bin/node_exporter

[Install]

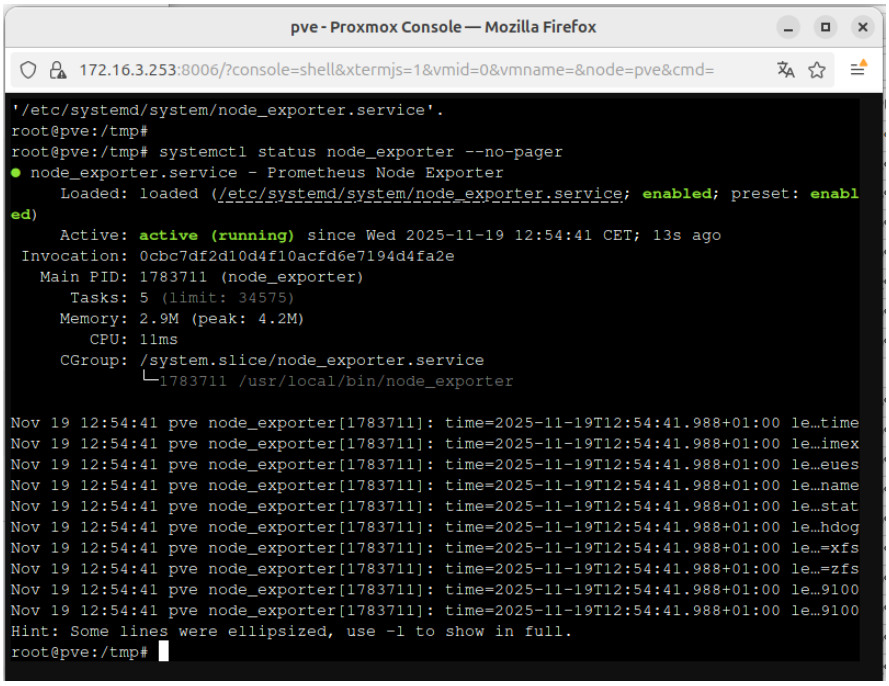
WantedBy=multi-user.target

Activamos y comprobamos:

systemctl daemon-reload

systemctl enable --now node_exporter

systemctl status node_exporter –no-pager



AÑADIR PROXMOX COMO TARGET EN PROMETHEUS.

Salimos de proxmox y se vuelve al miniservidor, se edita el archivo prometheus.yml

nano /monitoring/prometheus/prometheus.yml

global:

scrape_interval: 15s

scrape_configs:

- job_name: "prometheus"

static_configs:

- targets: ["prometheus:9090"]

- job_name: "miniservidor"

static_configs:

- targets: ["192.168.20.21:9100"]

- job_name: "proxmox"

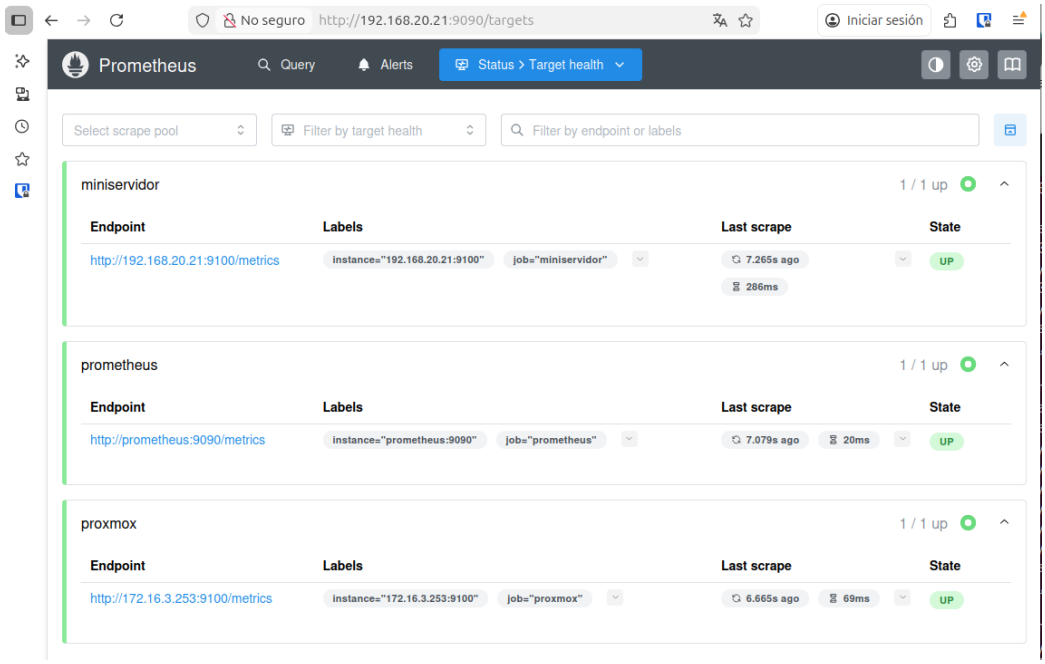
static_configs:

- targets: ["172.16.3.253:9100"]

Reiniciamos Prometheus:

docker compose restart prometheus

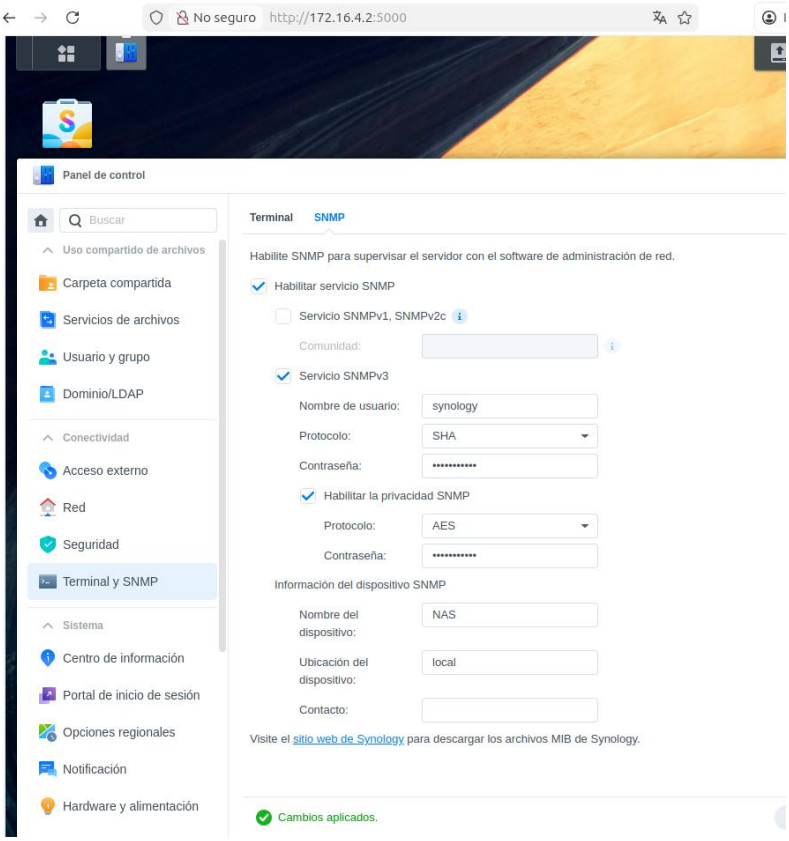
Y comprobamos en la web que aparecen los targets.



5.2.3. Monitorización NAS Synology.

Se habilita el protocolo SNMP en el Synology.

Se configura snmpv3 que es más seguro y lo recomendado por Synology y Prometheus también soporta snmpv3.



SNMP EXPORTER

Se creó dentro del proyecto un directorio específico para SNMP Exporter: ~/monitoring/snmp_exporter/

Fue necesario utilizar el generador oficial de Prometheus para que se incluyan todas las MIB que se necesitan de Synology.

El generador se descarga de github y se importan las mibs:

git clone https://github.com/prometheus/snmp_exporter.git

cd snmp_exporter/generator

Se añadió la MIB oficial de Synology (incluida en su documentación) dentro del directorio:

snmp_exporter/generator/mibs/synology/

Con esto el generados conoce todos los OID del NAS.

Se genera ahora de forma automática el snmp.yml

make clean

make mibs

make generate -> Este commando genera el archivo final: ~/monitoring/snmp_exporter/snmp.yml

Este archivo contiene todas las definiciones necesarias para que Prometheus pueda consultar el NAS con SNMPv3.

Integración del SNMP Exporter en Docker Compose

Dentro del archivo docker-compose.yml se añadió un nuevo servicio que ejecuta el contenedor oficial:

```
snmp-exporter:
  image: prom/snmp-exporter:latest
  container_name: snmp-exporter
  ports:
    - "9116:9116"
  volumes:
    - ./snmp_exporter/snmp.yml:/etc/snmp_exporter/snmp.yml:ro
  networks:
    - monitoring
  restart: unless-stopped
```

Configuración del job de Prometheus

Editamos el archivo de prometheus y añadimos el nuevo job.

En el caso del NAS Synology, el job de Prometheus no se puede configurar igual que los demás porque este dispositivo no entrega métricas por HTTP como hacen Prometheus, Node Exporter o Grafana. El NAS solo habla por SNMP, así que Prometheus no puede conectarse directamente a él.

Por eso necesitamos usar el SNMP Exporter, que es el que se encarga de recoger la información del NAS y convertirla en un formato que Prometheus pueda entender.

Esto hace que la configuración sea distinta: en targets no va la ip del NAS, sino snmp-exporter, la ip real del NAS se pasa como un parámetro "target". Para que todo esto funcione hay que usar las reglas de *relabel_configs* → Reescriben la dirección para que Prometheus construya bien la URL

- job_name: "synology"

metrics_path: /snmp

params:

module: [synology]

auth: [synology_v3]

static_configs:

- targets: ["172.16.4.2"]

relabel_configs:

- source_labels: [__address__]

target_label: __param_target

- source_labels: [__param_target]

target_label: instance

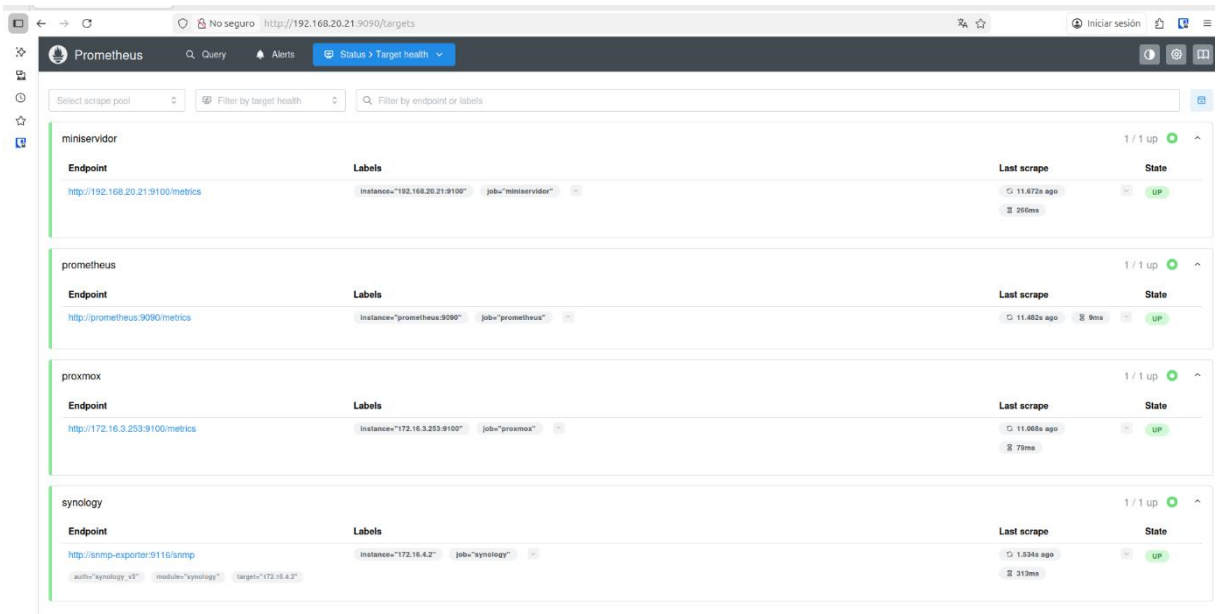
- target_label: __address__

replacement: "snmp-exporter:9116"

Reiniciamos Prometheus:

docker compose restart prometheus

y vemos en la interfaz web de prometheus que ya aparece:



5.2.4. Monitorización de contenedores LXC dentro de Proxmox

En este punto actualmente se está monitorizando:

- Proxmox, el nodo, con node_exporter: cpu,ram,disco,red,carga,uptime del host.

-NAS Synology con snmpv3_exporter: estado general, disco, etc.

En esta fase se procede a instalar Node Exporter en los LXC que nos interesan y a integrarlos en Prometheus.

Instalación de Node Exporter dentro de los contenedores LXC

En cada contenedor que se quiera monitorizar tiene que exponer las métricas, para ello se instala node_exporter en cada LXC, en este caso se instalarán en dos de ellos.

El procedimiento es el mismo que cuando se instaló en el nodo PROXMOX.

Dentro de cada contenedor:

cd /tmp

curl -LO
https://github.com/prometheus/node_exporter/releases/download/v1.8.2/node_exporter-1.8.2.linux-amd64.tar.gz

tar xzf node_exporter-1.8.2.linux-amd64.tar.gz

mv node_exporter-1.8.2.linux-amd64/node_exporter /usr/local/bin/

useradd -rs /bin/false node_exporter

Se añade un servicio systemd para asegurar su ejecución continua:

Nano /etc/systemd/system/node_exporter.service

[Unit]

Description=Prometheus Node Exporter

Wants=network-online.target

After=network-online.target

[Service]

User=node_exporter

Group=node_exporter

ExecStart=/usr/local/bin/node_exporter

[Install]

WantedBy=multi-user.target

Se active:

systemctl daemon-reload

systemctl enable --now node_exporter

En en miniservidor:

Se modifica el archivo de prometheus.yml y se añade el job correspondiente a cada lxc:

- job_name: "lxc-dokuwiki"

  static_configs:

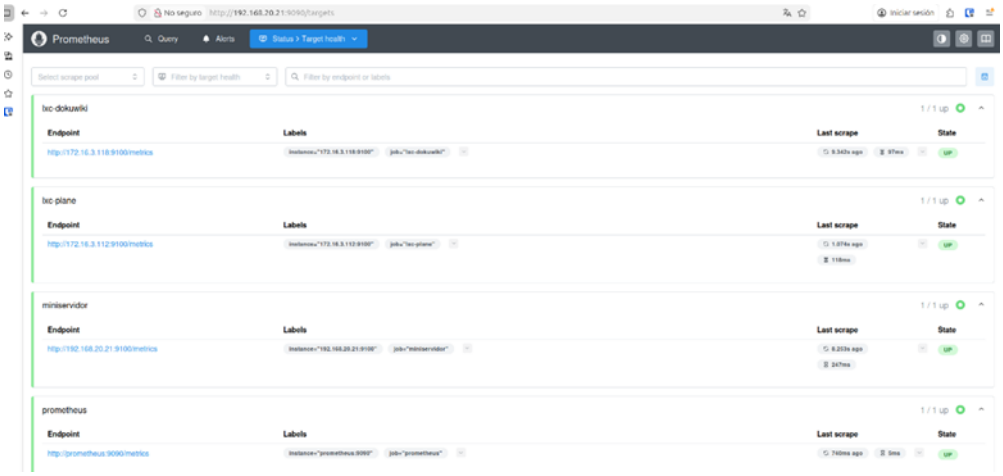
    - targets: ["172.16.3.118:9100"]

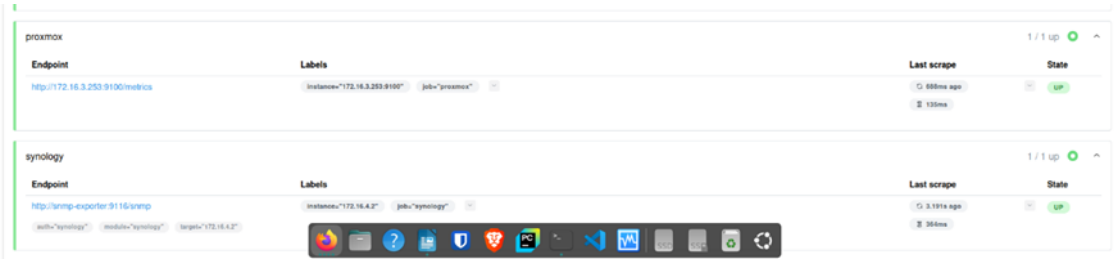
Y finalmente se recarga prometheus:

docker compose down prometheus

docker compose up -d Prometheus

Verificamos todos los targets cargados correctamente:





Integración de Blackbox Exporter para verificación de accesibilidad

En este punto actualmente se está monitorizando:

- **Proxmox**, el nodo principal, mediante *node_exporter* (CPU, RAM, disco, red, carga, uptime).
- **NAS Synology**, mediante *snmp_exporter* con SNMPv3 (estado general, discos, temperatura, IOPS, etc.).
- **Contenedores LXC**, instalando *node_exporter* dentro de cada uno para obtener métricas completas de su rendimiento.

Además de la monitorización interna mediante Node Exporter, se añadió Blackbox Exporter como herramienta complementaria para comprobar la accesibilidad externa de servicios alojados en los contenedores.

Blackbox permite realizar comprobaciones como:

- Ping/ICMP
- HTTP/HTTPS (respuesta web del contenedor)
- TCP (puertos expuestos por un servicio)

Se incorpora el contenedor Blackbox modificando el archivo de docker-compose.yml

```
blackbox:
  image: prom/blackbox-exporter:latest
  container_name: blackbox
  ports:
    - "9115:9115"
  networks:
    - monitoring
  restart: unless-stopped
```

Y se añade el job en el archivo de prometheus.yml:

```
- job_name: "blackbox-lxc"
  metrics_path: /probe
  params:
    module: [icmp]
  static_configs:
    - targets:
        - 172.16.3.118  # LXC Dokuwiki
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - target_label: __address__
      replacement: blackbox:9115
```

Se crea archivo de blacbox:

Se crea una carpeta dentro de monitoring.

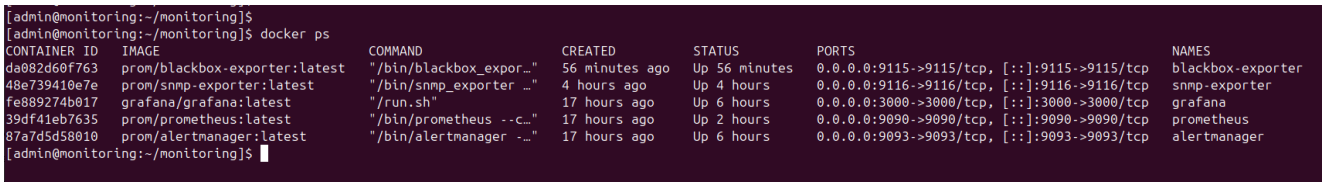
mkdir -p ~/monitoring/blackbox

Se genera el archivo config.yml

```
[admin@monitoring:~/monitoring/blackbox]$ cat config.yml
modules:
  icmp:
    prober: icmp
    timeout: 5s
    icmp:
      preferred_ip_protocol: ip4

  http_2xx:
    prober: http
    timeout: 5s
    http:
      valid_status_codes: []
      method: GET

  tcp_connect:
    prober: tcp
    timeout: 5s
[admin@monitoring:~/monitoring/blackbox]$
```

LEVANTAR CONTENEDOR BLACKBOX

Selevanta el contenedor con: docker compose up -d blackbox-exporter.



5.2.5. Dashboards Grafana para Proxmox, NAS y LXC.

Con todas las métricas ya disponibles en Prometheus, se diseña un dashboard unificado en Grafana denominado “Monitorización – Vista general”.
Este dashboard permite consultar de forma centralizada el estado del nodo Proxmox, los contenedores LXC y el NAS Synology.

El dashboard se divide en rows:

- Proxmox – Estado general
- Nas
- Contenedores LXC

En cada fila se añaden paneles con las métricas que interesan monitorizar

Las consultas se realizaron en PromQL. A continuación se muestran algunos ejemplos utilizados:

CPU Proxmox % -> 100 - (avg by(job) (irate(node_cpu_seconds_total{job="proxmox",mode="idle"}[5m])) * 100)

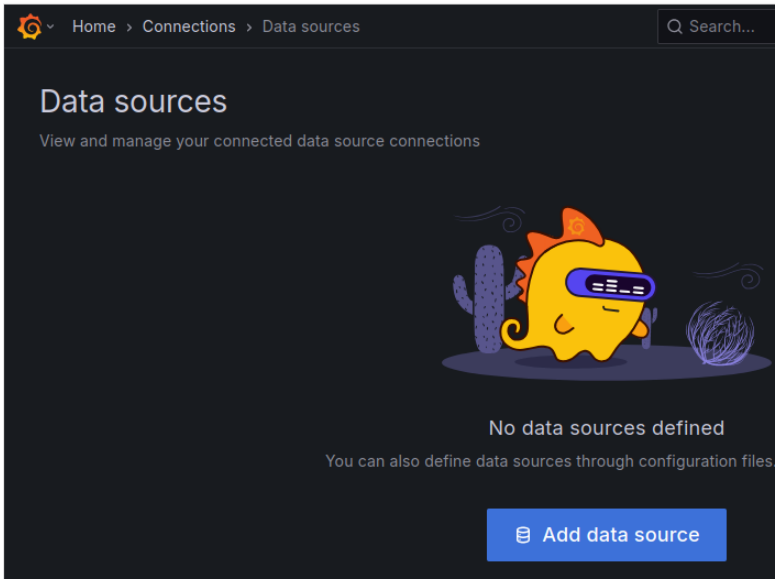
Espacio libre en disco (gb) -> 100 - (avg by(job) (irate(node_cpu_seconds_total{job="proxmox",mode="idle"}[5m])) * 100)

RAM usada (%) -> (node_memory_MemTotal_bytes{job="proxmox"} - node_memory_MemAvailable_bytes{job="proxmox"})

/ node_memory_MemTotal_bytes{job="proxmox"} * 100

Añadir Prometheus como fuente de datos en Grafana

En el navegador de grafana: home/connections/data source.



Se pulsa sobre Connections - data source – add data source y se selecciona prometheus.

En URL se le indica: http://prometheus:9090

Esto es porque al estar dentro de la misma red docker, se pone el nombre del contenedor y no la ip.

No se modifica nada más y se guarda Save & test.

Se procede a crear el Dashboard:

En el menú izquierdo: dashboards - new dashboards - add visualization - data source: prometheus

Le asignamos como nombre: Monitorización – Vista general

Aquí no cubrimos nada más y le damos a guardar.

Una vez creado el Dashboard se edita y organiza.

Se crean cuatro *rows* (Proxmox, NAS, LXC-dokuwiki y LXC-plane). Dentro de cada fila se añaden los paneles con las métricas que nos interesen.

Para evitar repetir el mismo proceso con todas las métricas, se muestra a continuación un ejemplo completo de cómo se crea un panel en Grafana y cómo se construye una query en PromQL. El resto de paneles siguen exactamente la misma metodología.

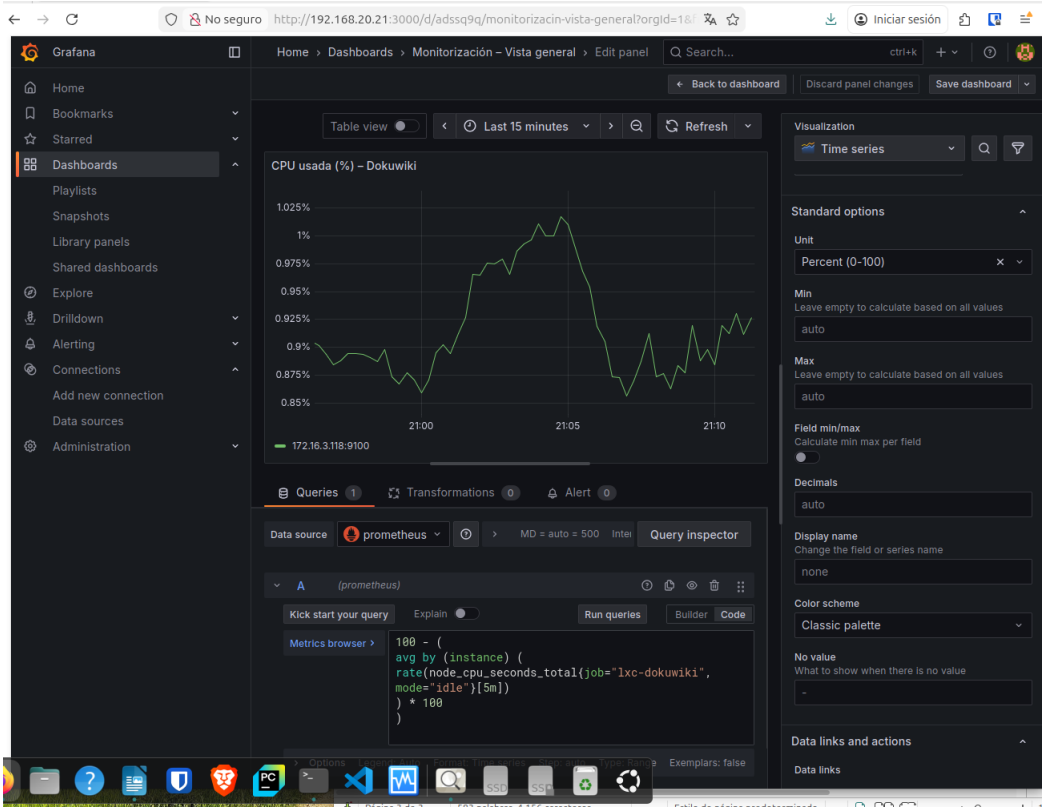
Ej. Para el panel de “CPU usada (%)” de LXC-docuwiki

En el panel principal de grafana creado: Monitorización – Vista general

Editar dashboard – add – visualization.

Abajo en query se introduce la siguiente expresión:

```
100 - (
avg by (instance) (
rate(node_cpu_seconds_total{job="lxc-dokuwiki", mode="idle"}[5m])
) * 100
)
```



Explicación de la query:

- node_cpu_seconds_total es la métrica que expone el node_exporter de cada contenedor.
- mode="idle" indica que se está midiendo el tiempo que la CPU estuvo sin hacer trabajo.
- rate(...[5m]) calcula la velocidad de cambio en los últimos 5 minutos.
- avg by (instance) agrupa el cálculo por contenedor.
- Se hace 100 - idle% para obtener el porcentaje real de CPU usada.

El panel tiene diferentes tipos de configuración, en este caso se le aplica

- Tipo de visualización: Time series
- Unidad: Percent (0–100)
- Thresholds opcionales:

    -Verde: < 60%

    -Amarillo: 60–85%

    -Rojo: > 85%

PANELES NAS

Para el NAS Synology se ha usado snmp_exporter con SNMPv3, generando una configuración específica a partir de la MIB oficial de Synology. Estos se han descargado directamente de la web de Synology.

Las métricas disponibles dependen exclusivamente de lo que expone el propio Synology a través de SNMP, por lo que se ha trabajado con las métricas realmente presentes en el snmp.yml configurado. Entre ellas destacan:

Estado general del NAS:

up{job="synology"}

Uso de CPU (%):

100 - avg by (instance) (ssCpuIdle{job="synology"})

A continuación vemos el dashboard creado organizado por rows

5.2.6. ALERTMANAGER

Como primer paso se configura prometheus para cargar un archivo externo de reglas, que crearemos: alerts.yml.

En este archivo se definen todas las reglas que se quieran implementar.

Archivo alerts.yml

groups:

- name: general-rules

rules:

En el archivo de prometheus.yml se enlaza este archivo creado y se le indica que lo envíe a alertmanager:

líneas que se añaden al archivo:

rule_files:

- "alerts.yml"

alerting:

alertmanagers:

- static_configs:

- targets: ["alertmanager:9093"]

En Alertmanager se configura para usar servidor smtp de gmail (se creó un correo ficticio para las pruebas que posteriormente se eliminará). Si se utiliza gmail es necesario utilizar contraseña de aplicación.

Archivo de alertmanager.yml:

global:

smtp_smarthost: 'smtp.gmail.com:587'

smtp_from: '86monitoring@gmail.com'

smtp_auth_username: '86monitoring@gmail.com'

smtp_auth_password: 'CONTRASEÑA_DE_APLICACIÓN'

route:

receiver: 'email-alerts'

receivers:

- name: 'email-alerts'

email_configs:

- to: '86monitoring@gmail.com'

Procedemos a poner de ejemplo una.

Alarma contenedor caído, ej: LXC DokuWiki: 172.16.3.118

Todas las alertas se configuran en el archivo alerts.yml dentro del grupo general-rules:

Archivo alerts.yml

groups:

- name: general-rules

rules:

- alert: LXCDown

expr: up{instance="172.16.3.118:9100"} == 0

for: 1m

labels:

severity: critical

annotations:

summary: "Contenedor LXC DokuWiki caído"

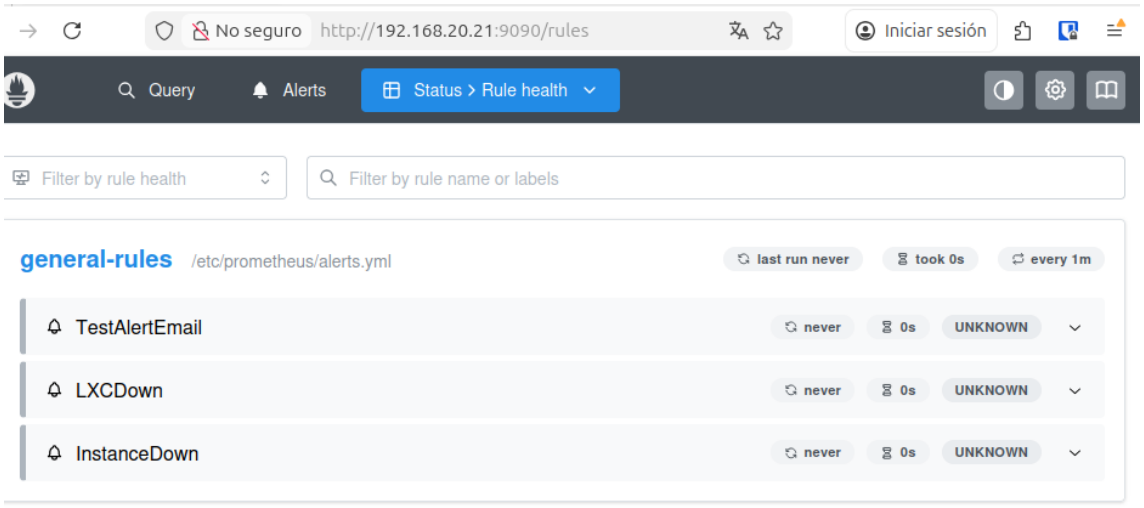
description: "El contenedor en 172.16.3.118 no responde desde hace más de 1 minuto."

Tras guardar el archivo:

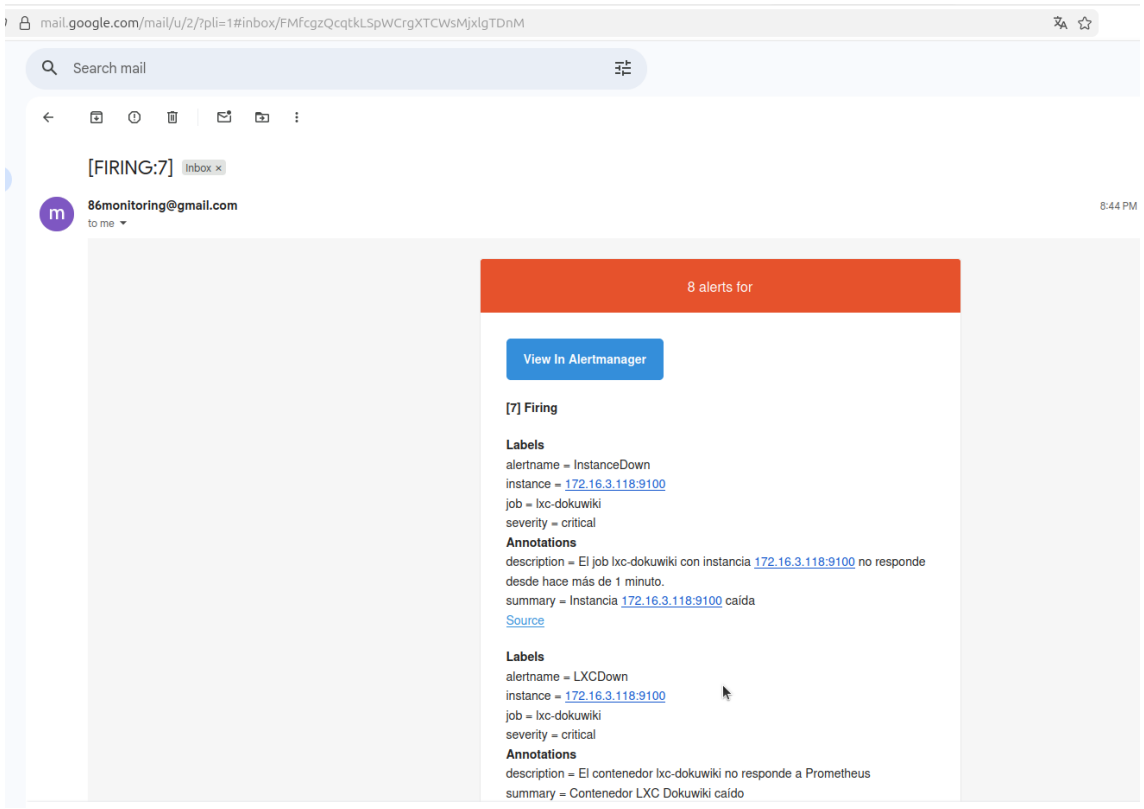
docker compose down prometheus

docker compose up -d prometheus

En el navegador, en menú Status / Rules observamos la regla creada de LXCD



Para verificar que funciona, paramos ese contenedor y verificamos que avisa vía mail.



Tipos de alertas previstas en el sistema.

Es recomendable definir varios tipos de alertas para cubrir los puntos críticos del sistema.

1. **Alertas de disponibilidad**
    - Contenedor LXC caído.
    - Nodo Proxmox inaccesible.
    - NAS Synology no responde por SNMP.
2. **Alertas de rendimiento**
    - CPU del Proxmox por encima de un umbral.
    - RAM Proxmox o de un LXC por encima de un umbral.
    - Carga del sistema demasiado alta.
    - Uso de red excesivo o saturación.
3. **Alertas de almacenamiento**
    - Espacio en el NAS por debajo de un porcentaje.
    - Discos con sectores dañados o fallos SMART.
    - Volumen del Proxmox por encima de un umbral.
4. **Alertas de servicios (opcional)**
    - Servicio web del LXC no responde.
    - Servicio SSH en Proxmox inaccesible.
    - Cualquier endpoint monitorizado vía Blackbox Exporter.

5.2.7. NAS Synology como almacenamiento den Proxmox y volcado de backups

DESDE EL NAS

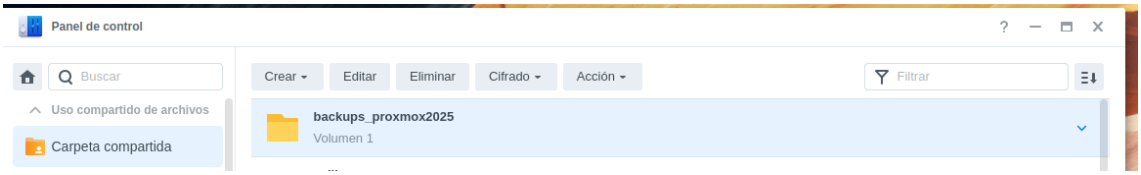
Es necesario la creación de una carpeta compartida en el NAS.

Como primer paso se crea un usuario al que se le dará permisos en esa carpeta.

Creamos carpeta *-> Panel de Control / Carpeta compartida / Crear*

Solo ese usuario tendrá acceso de lectura y escritura en esa carpeta.

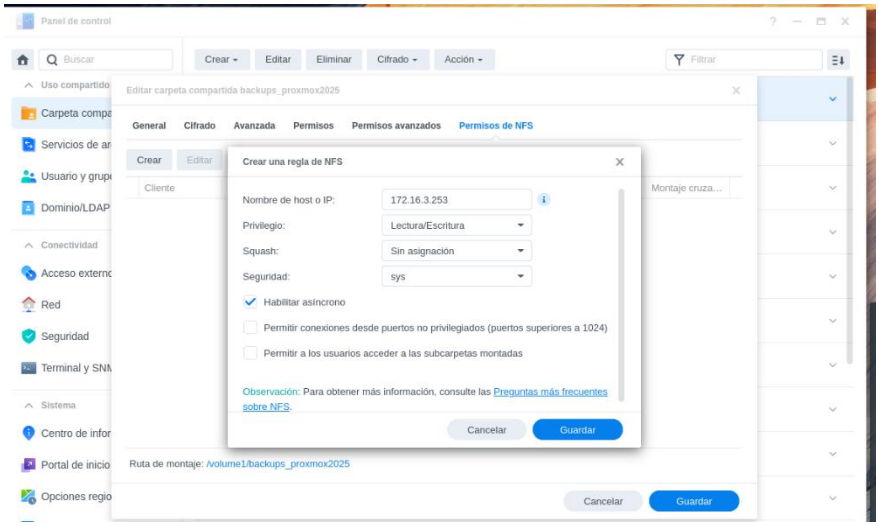
Creación carpeta compartida:



Importante revisar en *NAS / Panel de control / servicio de archivos*, que esté activado el servicio **NFS**.

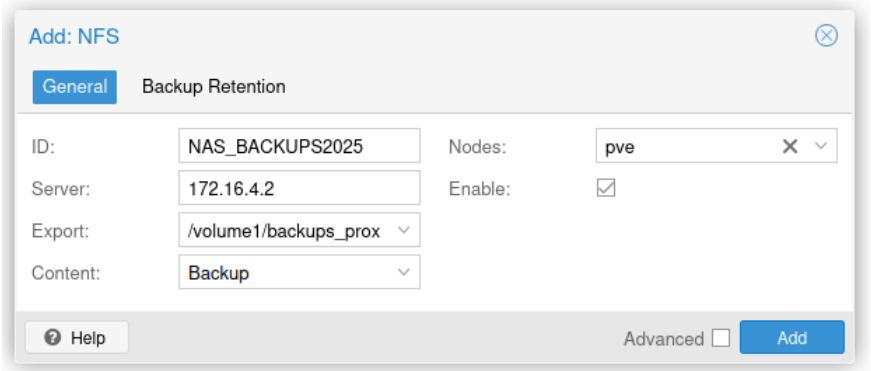
Es necesario activar este servicio porque NFS es el sistema recomendado por Proxmox para almacenamiento de backups, ofrece mejor rendimiento y soporta formatos VZDumps.

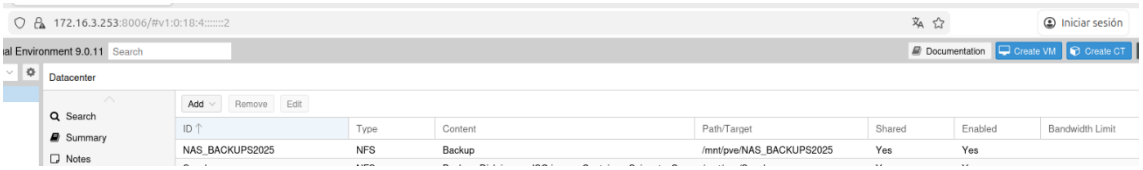
Volvemos a la carpeta compartida, editar / permisos de NFS / crear y ahí creamos la regla nfs con la IP del proxmox.



EN EL PROXMOX.

Desde el Proxmox: Datacenter / Storage / Add / NFS y cubrimos con los siguientes datos





CREAR POLÍTICA DE COPIAS AUTOMÁTICAS (bakup job)

En *Datacenter / Backup / add*

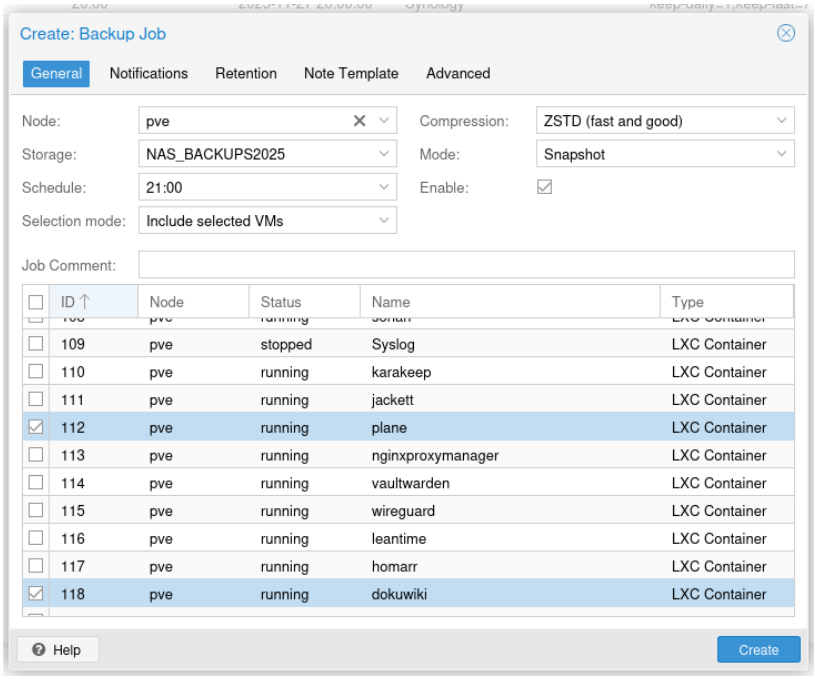
Cubrimos los datos que nos pide. En *selection mode*, nos permite seleccionar si queremos copia de todos los contenedores o solo de algunos que nos interesen.

En la pestaña de **Notifications** se introduce el mail.

En la pestaña **Retention** le indicamos keep last → 3

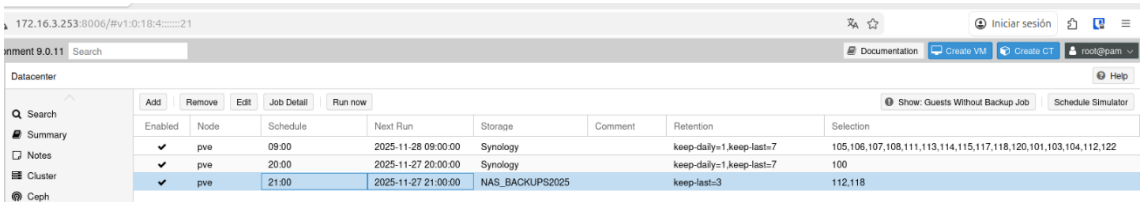
Eso es para que mantenga 3 copias antes de borrar las más antiguas.

Así no se acumulan demasiadas copias, se evita llenar el disco y tener que hacerlo de manera manual.

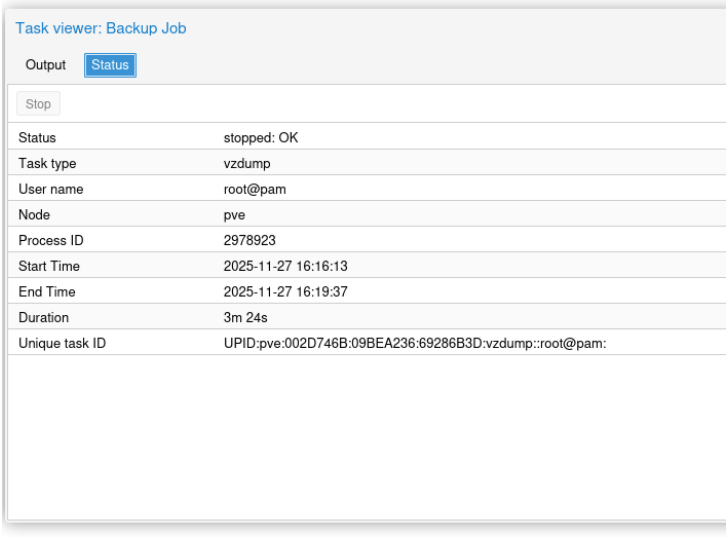


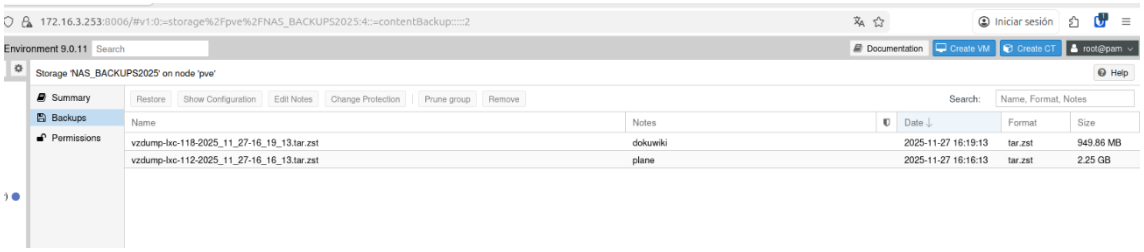
Se fuerza backup para prueba

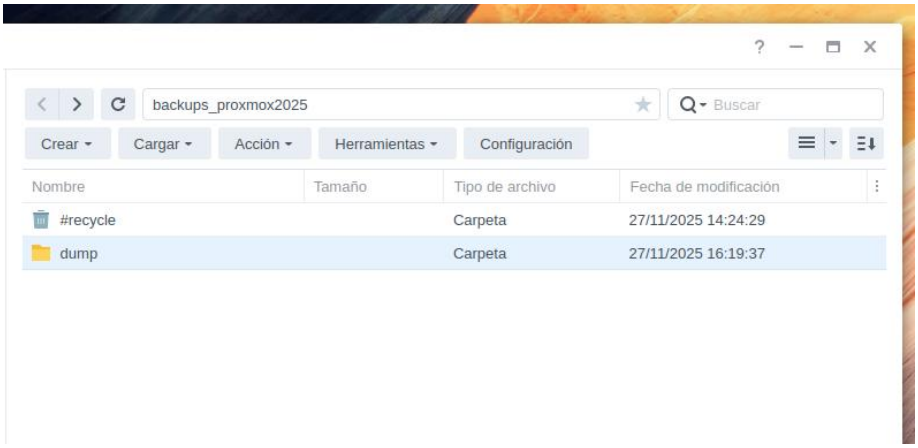
En *Datacenter / backup*, seleccionamos nuestro job y le damos a *RUN NOW*



Se comprueba que realizó correctamente la copia:





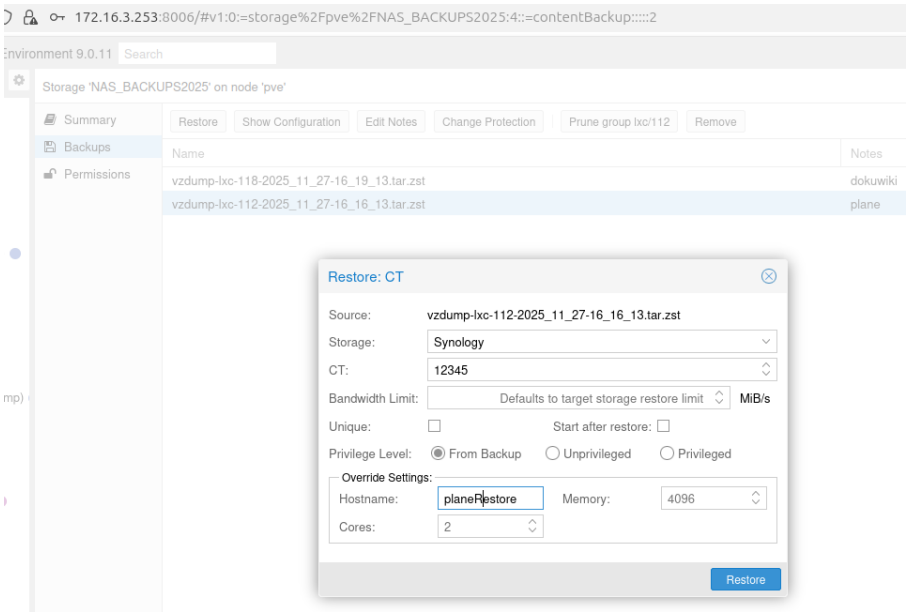


Restaurar el backup

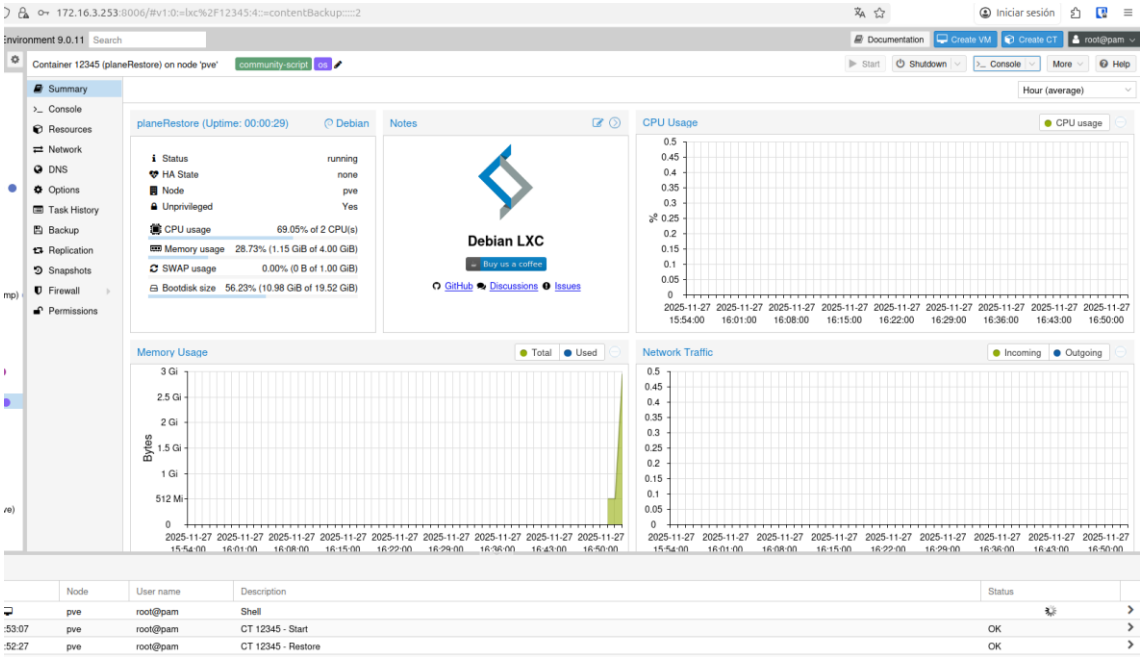
Desde storage, seleccionamos *NAS_BACKUP2025 / backups*

Seleccionamos uno de los backups, plane por ejemplo y pulsamos restore

En la pantalla que sale cubrimos los datos y ponemos un ct nuevo para que cree un lxc nuevo y no pisar el que ya tenemos.



Comprobamos que ya nos hizo la copia, lo levantamos y vemos que está corriendo correctamente.



6. Propuesta de mejoras

7. Conclusiones