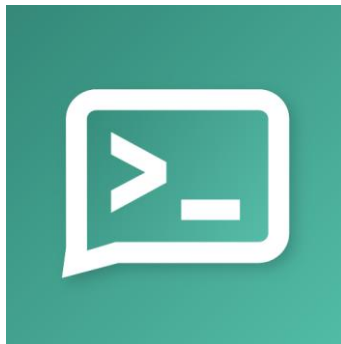


Trabajo Práctico

NTFY

Laboratorio de Redes y Sistemas Operativos - primer cuatrimestre de 2026



Integrantes:

- Cuevas, Leandro Agustín
- Moreira Toja, Ignacio
- Sofarelli, Thiago Ezequiel

Índice

Introducción	3
¿Qué es ntfy?	3
Nuestra implementación	3
Máquina utilizada	3
Software utilizado y/o necesario	3
Instalación y Configuración	4
Actualización del gestor de paquetes	4
Instalación de dependencias	4
Configuración de dependencias	4
Creación de base de datos PostgreSQL	4
Configuración de proxy inverso con nginx	4
Instalación de ntfy	5
Configuración de ntfy	5
Reinicio y verificación de status de servicios	6
Funcionamiento	6
Creación de usuarios	6
Lista de control de acceso (ACL)	7
Publicación de mensajes en un tópico	8
Suscripción sobre un tópico	9
Notificaciones por email	9
Soporte para archivos adjuntos (attachments)	9
Interfaz Web	11
Problemáticas	12
Conclusiones y uso de IA	13

Introducción

¿Qué es ntfy?

ntfy es un servicio de mensajería basado en el modelo publicador-suscriptor que utiliza peticiones HTTP para la comunicación en tiempo real. Este servicio permite que un usuario publique mensajes en un tópico y los suscriptores de dicho tópico reciban esa información en forma de notificación.

Nuestra implementación

Para este proyecto, desplegamos ntfy en un entorno privado, reemplazando el motor por defecto de SQLite por PostgreSQL para que la base de datos sea más robusta y escalable. Como servidor web, usamos Nginx actuando como un proxy inverso, lo que nos permite gestionar mejor el tráfico y las conexiones en tiempo real.

En cuanto a la seguridad, establecimos la lista de control de acceso (ACL) con el fin de crear usuarios específicos y asignarles permisos sobre distintos tópicos.

También configuramos el envío de archivos adjuntos y el envío de notificaciones vía SMTP para que los usuarios puedan recibir las notificaciones por email.

Máquina utilizada

OS: Ubuntu 24.04.4 LTS x86_64

Host: VirtualBox 1.2

Kernel: 6.17.0-35-generic

Shell: bash 5.2.21

CPU: 12th Gen Intel i7-1255U (4) @ 2

GPU: 00:02.0 VMware SVGA II Adapter

Memory: 1285MiB / 3914MiB

Software utilizado y/o necesario

Servicio: ntfy, utilizado como broker de mensajes y encargado de la arquitectura publicador-suscriptor.

Base de datos: PostgreSQL, utilizado para el almacenamiento persistente y escalable de la información del sistema.

Proxy: nginx, software de código abierto que actúa como servidor web y proxy inverso.

Herramientas adicionales: wget, gestor de paquetes apt, gestor de servicios systemd.

Instalación y Configuración

Actualización del gestor de paquetes

Actualizamos el gestor de paquetes y aplicamos dichas actualizaciones

```
sudo apt update && sudo apt upgrade -y
```

Instalación de dependencias

Instalamos las dependencias a utilizar: nginx y postgresql

```
sudo apt install postgresql nginx -y
```

Habilitamos las dependencias y verificamos que estén funcionando

```
sudo systemctl enable nginx postgresql
```

```
sudo systemctl start nginx postgresql
```

```
sudo systemctl status nginx postgresql
```

Configuración de dependencias

Creación de base de datos PostgreSQL

Abrimos la consola interactiva de postgres con el usuario por default

```
sudo -u postgres psql
```

Creamos un usuario

```
CREATE USER ntfy WITH PASSWORD '1234';
```

Creamos la base de datos ntfy_db, siendo el usuario anteriormente creado el dueño de la misma

```
CREATE DATABASE ntfy_db OWNER ntfy;
```

Salimos de la consola interactiva

```
\q
```

Configuración de proxy inverso con nginx

Debemos crear el sitio de ntfy con la configuración especificada para que nginx intercepte las peticiones HTTP del puerto 80 y las redirija hacia el servicio ntfy, el cual se encuentra escuchando en el puerto 2586:

```
sudo nano /etc/nginx/sites-available/ntfy
```

```
server {
    listen 80;
    server_name [IP];
    location / {
        proxy_pass http://127.0.0.1:2586;
        proxy_http_version 1.1;
        proxy_buffering off;
        proxy_request_buffering off;
    }
}
```

```

    proxy_redirect off;
    proxy_set_header Host $http_host;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_connect_timeout 3m;
    proxy_send_timeout 3m;
    proxy_read_timeout 3m;
    client_max_body_size 0;
}
}

```

Guardamos el archivo y chequeamos la sintaxis

```
sudo nginx -t
```

Eliminamos el sitio por default que muestra nginx

```
sudo rm /etc/nginx/sites-enabled/default
```

Enlazamos el archivo que creamos anteriormente para que nginx cargue la web de ntfy

```
sudo ln -sf /etc/nginx/sites-available/ntfy /etc/nginx/sites-enabled/
```

Instalación de ntfy

Descargamos el archivo .deb

```
wget
```

```
https://github.com/binwiederhier/ntfy/releases/download/v2.26.0/ntfy_2.26.0_linux_amd64.deb
```

Procedemos a la instalación

```
sudo dpkg -i ntfy_*.deb
```

Habilitamos ntfy y verificamos que esté funcionando

```
sudo systemctl enable ntfy
```

```
sudo systemctl start ntfy
```

```
sudo systemctl status ntfy
```

Configuración de ntfy

Debemos modificar el archivo server.yml con el fin de activar el proxy y restringir la escucha de ntfy al entorno local, realizar la conexión de ntfy a nuestra base de datos postgres, desactivar los accesos por defecto (pues luego utilizaremos ACL), configurar el directorio donde se guardarán los archivos adjuntos, la duración de los mensajes en caché, y el servicio SMTP para luego poder recibir notificaciones por email:

```
sudo nano /etc/ntfy/server.yml
```

```
base-url: "http://[IP]" # IP del servidor
```

```
listen-http: "127.0.0.1:2586"
```

```
behind-proxy: true
```

```
database-url: "postgres://ntfy:1234@localhost:5432/ntfy_db" # Datos de la DB
creada anteriormente
auth-default-access: "deny-all" # Desactivación de accesos por default
enable-login: true # Esto es opcional: permite iniciar sesión desde la web.
enable-signup: true # Esto es opcional: permite registrarse desde la web.
attachment-cache-dir: "/var/cache/ntfy/attachments" # Carpeta de archivos
adjuntos
cache-duration: "1h" # Duración de mensajes en caché
smtp-sender-addr: "smtp.example.com:587" # Datos SMTP
smtp-sender-from: "ntfy@example.com" # Datos SMTP
smtp-sender-user: "yoursmtpuser" # Datos SMTP
smtp-sender-pass: "yoursmtppass" # Datos SMTP
```

Creamos el directorio para guardar los archivos adjuntos

```
sudo mkdir -p /var/cache/ntfy/attachments
```

Le otorgamos los permisos de owner al usuario ntfy

```
sudo chown -R ntfy:ntfy /var/cache/ntfy
```

Aclaraciones:

- Los parámetros definidos en el archivo `server.yml` fueron validados mediante la herramienta oficial de la web de `ntfy.sh`: <https://docs.ntfy.sh/config/>.
- Para hacer posible el envío de notificaciones por email, integramos el protocolo SMTP de gmail configurando las credenciales de autenticación necesarias.

Reinicio y verificación de status de servicios

Siendo que realizamos cambios en los archivos de configuración, debemos reiniciar los servicios. Para esto, ejecutamos:

```
sudo systemctl restart ntfy postgresql nginx
```

Ahora verificamos que los servicios estén activos y corriendo sin errores:

```
sudo systemctl status ntfy postgresql nginx
```

Funcionamiento

Creación de usuarios

ntfy provee dos roles: user y admin.

El rol user no otorga permisos, mientras que el rol admin otorga permisos de lectura y escritura hacia todos los tópicos.

Para crear un usuario con rol user, ejecutamos el siguiente comando:

```
ntfy user add nombreDeUsuario
```

Para crear un usuario con rol admin, ejecutamos el siguiente comando:

```
ntfy user add --role=admin nombreDeUsuario
```

Una vez ejecutado cualquiera de los dos comandos anteriores, se debe escribir la password del usuario en la consola y darle Enter.

Para listar dichos usuarios, ejecutamos el siguiente comando:

```
ntfy user list
```

Esto da como respuesta el listado de usuarios junto con sus roles. A modo de ejemplo:

```
user phil (role: admin, tier: none)
- read-write access to all topics (admin role)
user thiago (role: admin, tier: none)
- read-write access to all topics (admin role)
user lean (role: user, tier: none)
- no topic-specific permissions
```

También es posible modificar el rol de un usuario. A modo de ejemplo, si queremos que el usuario “leandro” posea el rol admin, ejecutamos:

```
ntfy user change-role leandro admin
```

Lista de control de acceso (ACL)

La lista de control de acceso administra los accesos a los tópicos para aquellos usuarios no administradores, y para el acceso anónimo.

Para administrar la lista de control de acceso, debe ejecutarse el siguiente comando, encargado de otorgarle permisos específicos a un usuario para un tópico determinado:

```
ntfy access nombreDeUsuario tópico permiso
```

donde **nombreDeUsuario** es el usuario al cual se le asignarán los permisos (aclaración: si se trata de acceso anónimo, debe usarse **everyone**), **tópico** es el tópico sobre el cual dicho usuario obtendrá los permisos, y **permiso** es el permiso a asignar.

Existen distintos tipos de permisos. Ellos son:

read-write (permite que el usuario publique mensajes en un tópico y también pueda suscribirse)

read-only (permite que el usuario se suscriba a un tópico)

write-only (permite que el usuario publique mensajes en un tópico)

deny (elimina todos los permisos)

Para listar los accesos de un usuario en específico, se debe ejecutar el siguiente comando:

```
ntfy access nombreDeUsuario
```

Probemos con **nombreDeUsuario** “nacho”. Esto da como resultado los accesos que posee dicho usuario. A modo de ejemplo:

```
user nacho (role: user, tier: none)
- read-write access to topic lenguajes
- read-only access to topic labo
```

Publicación de mensajes en un tópico

Antes de comenzar con la publicación de mensajes, hay que entender que los tópicos son creados en el momento en el que se envía un mensaje. Es decir, los tópicos no se crean de forma previa.

El formato estándar para utilizar ntfy CLI es `ntfy publish [OPCIONES] [TOPICO] [MENSAJE]`. Las opciones incluyen:

```
-u user:password # Autenticación de usuario
-t "title" # Título del mensaje
-p nivelDePrioridad # Indica el nivel de prioridad. Es numérico.
-e "email" # Esto envía el mensaje por mail. Por ahora, solo soporta título. Es decir, en el mail recibido no se ve nivel de prioridad ni archivos adjuntos.
-f=filepath # Archivo adjunto local
-a="attachment" # Archivo adjunto con URL externa
```

Para publicar un mensaje de forma anónima en un tópico usando **ntfy CLI**, ejecutamos el siguiente comando:

```
ntfy publish http://[IP]/tópico "Acá va el mensaje..."
```

A modo de ejemplo, podemos publicar el mensaje "Probando" en el tópico **labo** ejecutando el siguiente comando: `ntfy publish http://[IP]/labo "Probando"`

Esto publica el mensaje "Probando" en el tópico **labo**, y nos retorna información sobre el mensaje publicado en formato json:

```
{"id":"UEwK1qRkFctE","time":1784066650,"expires":1784070250,"event":"message","topic":"labo","message":"Probando"}
```

*Aclaración: por defecto, nuestro servicio no permite la publicación de mensajes anónimos. Es por esto que antes de publicar dicho mensaje, ejecutamos `ntfy access everyone labo read-write` para asignarle permisos de escritura y lectura sobre el tópico **labo** al usuario anónimo.*

Si un usuario sin permisos de escritura intenta publicar un mensaje en un tópico, lo hará de esta manera (probamos con el usuario nacho, con password nacho, y sin permisos de escritura sobre el tópico labo):

```
ntfy publish \
-u nacho:nacho \
http://[IP]/labo \
"Probando"
```

Obtendrá como resultado una respuesta json con código de error 403:

```
{"code":40301,"http":403,"error":"forbidden","link":"https://ntfy.sh/docs/publication/#authentication"}
```

Sin embargo, si le asignamos permisos de escritura (write-only) sobre ese tópico ejecutando el comando `ntfy access nacho labo write-only` y publica un nuevo mensaje, este se publicará de forma exitosa, obteniendo como resultado una respuesta json:

```
{"id":"TCCm0xwW4wRc","time":1784067382,"expires":1784070982,"event":"message","topic":"labo","message":"Probando"}
```

Suscripción sobre un tópico

Para que un usuario se suscriba a un tópico y vea los mensajes que se publican, debe poseer permisos de lectura. Una vez obtenidos, se debe ejecutar el siguiente comando:

```
ntfy sub -u user:pass http://[IP]/tópico
```

Vamos a asignarle permisos de lectura al usuario nacho `ntfy access nacho labo read-only` y ahora procedemos a suscribirnos ejecutando el siguiente comando:

```
ntfy sub -u nacho:nacho http://[IP]/labo.
```

De esta forma, al publicarse mensajes en el tópico **labo**, el usuario nacho los visualizará de la siguiente manera:

```
{"id":"j5DeI9qmmppY","time":1784068781,"expires":1784072381,"event":"message","topic":"labo","message":"Hola Nacho!"}
```

```
{"id":"6gqQyaT4FuJZ","time":1784068798,"expires":1784072398,"event":"message","topic":"labo","message":"Este es un recordatorio para que te hagas un café."}
```

Notificaciones por email

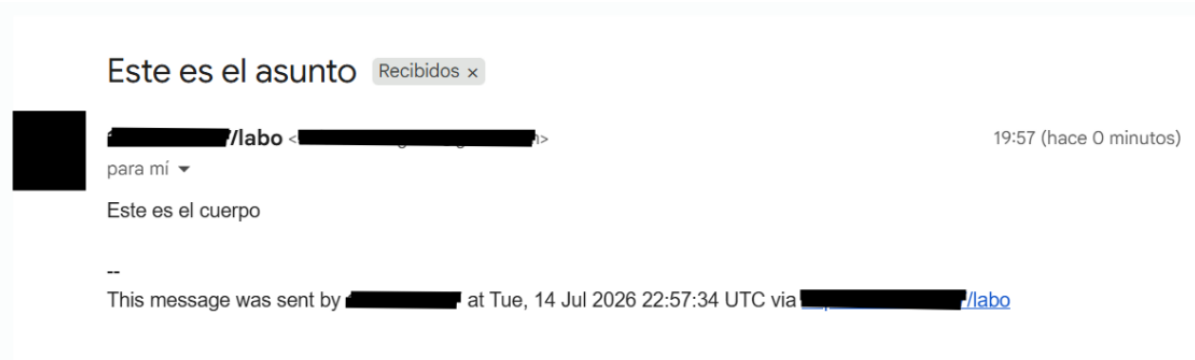
Es posible utilizar la opción `-e "email"` para enviar un mensaje y que el receptor obtenga la notificación por correo electrónico. Para esto, se debe configurar el servicio SMTP en el archivo `server.yml` como comentamos en la sección *Configuración de ntfy*.

A modo de ejemplo, haremos que el usuario Nacho envíe un mensaje con asunto y cuerpo a la dirección de correo electrónico ntfylabo@gmail.com y al tópico **labo**.

Para esto, se debe ejecutar el siguiente comando:

```
ntfy publish -u nacho:nacho -t "Este es el asunto" -e "ntfylabo@gmail.com" http://[IP]/labo "Este es el cuerpo"
```

De esta forma, la casilla de correo electrónico ntfylabo@gmail.com recibirá el siguiente correo:



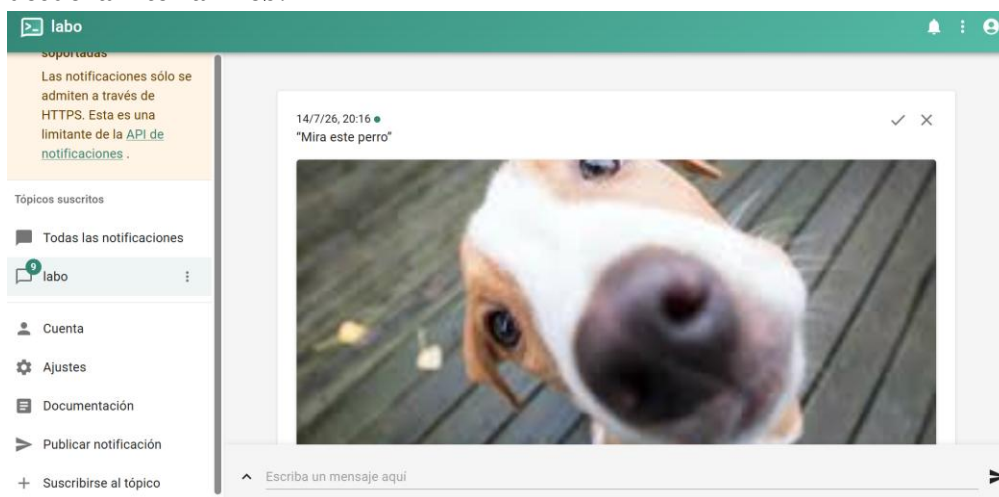
Soporte para archivos adjuntos (attachments)

Es posible utilizar la opción `-f filepath` o `-a attachment` para enviar un mensaje con un archivo adjunto. Para esto, se debe de crear el directorio que contendrá dichos archivos y se debe de especificar en el archivo `server.yml` como comentamos en la sección *Configuración de ntfy*.

Archivo local: A modo de ejemplo, haremos que el usuario Thiago envíe un mensaje que contenga el archivo local del path `"/home/vboxuser/imagenes/perro1.jpeg"` al tópico **labo**. Para esto, se debe ejecutar el siguiente comando:

```
ntfy publish -u thiago:thiago -f=/home/vboxuser/imagenes/perro1.jpeg  
http://[IP]/labo "Mira este perro"
```

Quienes estén suscritos a dicho tópico y tengan permisos de escritura, podrán ver la imagen desde la interfaz web:

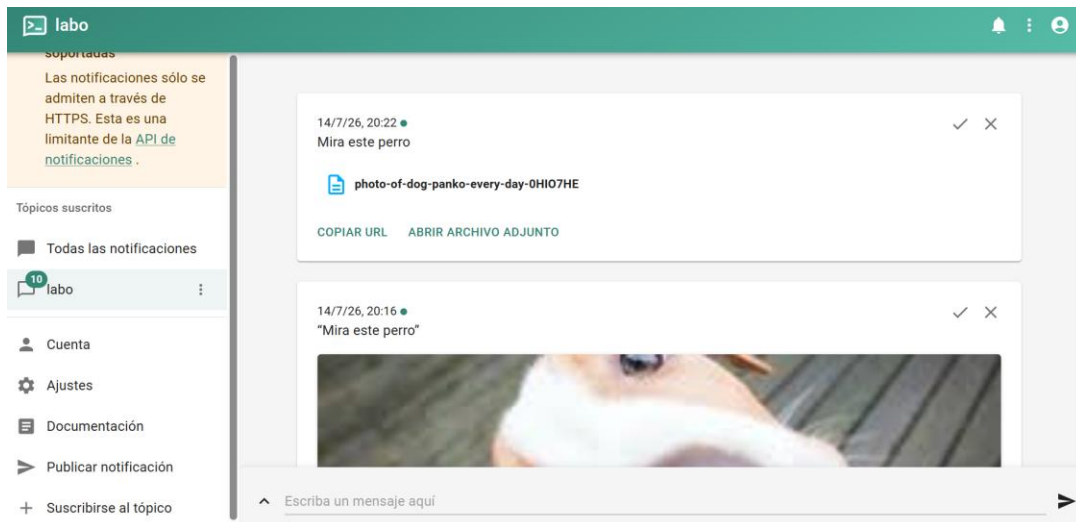


URL externa: A modo de ejemplo, haremos que el usuario Thiago envíe un mensaje que contenga el archivo externo <https://imgur.com/gallery/photo-of-dog-panko-every-day-0HIO7HE#/t/dog> al tópico **labo**.

Para esto, se debe ejecutar el siguiente comando:

```
ntfy publish -u thiago:thiago -a="https://imgur.com/gallery/photo-of-dog-panko-every-day-0HIO7HE#/t/dog" http://[IP]/labo "Mira este perro"
```

Quienes estén suscritos a dicho tópico y tengan permisos de escritura, podrán ver el archivo adjunto desde la interfaz web:

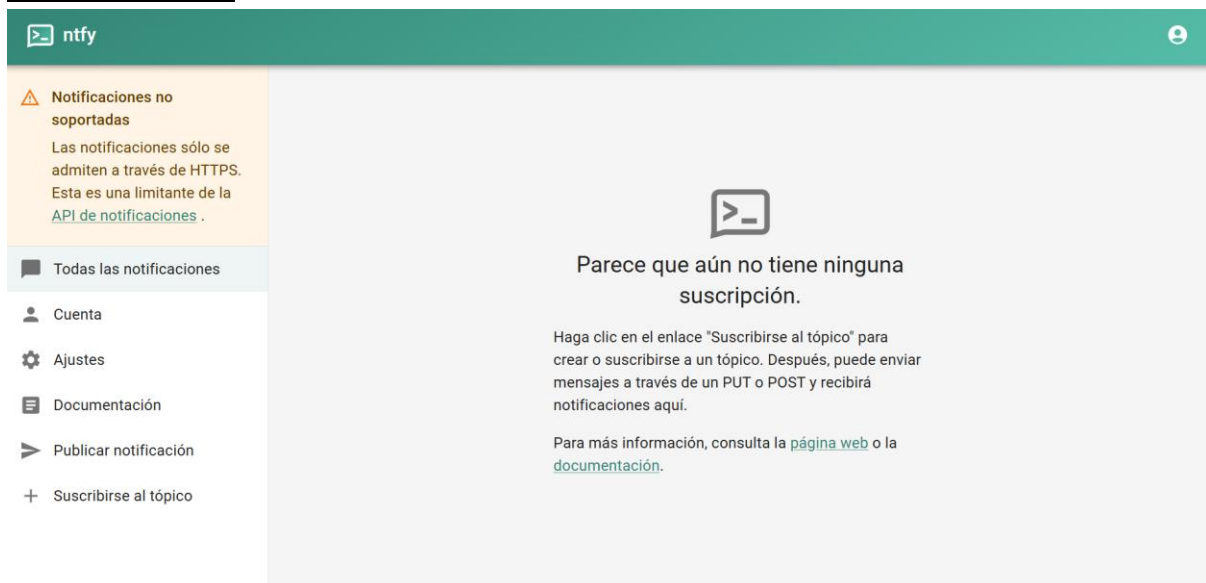


Interfaz Web

Podemos acceder a la interfaz web de ntfy desde `http://[IP]` y realizar distintas acciones. Entre ellas:

- loguearse/registrarse
- publicar mensajes en un tópico.
- suscribirse a un tópico y recibir notificaciones.
- administrar la cuenta (cambiar clave, email, etc).
- monitorear el uso (mensajes y correos enviados, entre otros).

Pantalla inicial



Monitoreo

Uso

Mensajes publicados ⓘ	3	de 17.280
Correos enviados ⓘ	0	de 24
Almacenamiento de archivos adjuntos <i>15 MB por archivo, eliminado después de 3 horas</i>	0 bytes	de 100 MB

Las estadísticas de uso y los límites de esta cuenta se basan en su dirección IP, por lo que podrían ser compartidos con otros usuarios. Los límites mostrados anteriormente son aproximados basados en los límites existentes.

Publicación de mensaje en un tópico

Publicar en [IP]/labo

URL del servicio	Nombre del tópico
http://[IP]	labo
Título	
Alerta de ejemplo	
Mensaje	
Este es un mensaje de ejemplo	

☐ Formatear como Markdown

😊 Etiquetas

Prioridad predeterminada ▼

Otras características:

Click URL

Reenviar al email

Adjuntar un archivo por URL

Adjuntar archivo local

Retraso en la entrega

☐ Publicar otro CANCELAR ENVIAR

Vista de mensajes sobre un tópico suscrito

labo

soportadas

Las notificaciones sólo se admiten a través de HTTPS. Esta es una limitante de la [API de notificaciones](#).

Tópicos suscritos

Todas las notificaciones

labo

Cuenta

Ajustes

14/7/26, 20:33

Alerta de ejemplo

Este es un mensaje de ejemplo

Problemáticas

Si bien la documentación oficial de ntfy detalla de forma bastante clara cómo implementar el servicio, surgieron algunos problemas:

- Siendo que debíamos implementar ACL, configuramos el *server.yml* con el código `auth-file: "/var/lib/ntfy/auth.db`. Sin embargo, cuando vimos el status de ntfy, el mismo decía **FAILED**. El log resultante decía lo siguiente: `if database-url is set, auth-file, cache-file, and web-push-file must not be set`. Esto se debe a que, como no usamos la DB por defecto de SQLite, no debíamos de especificar un `auth-file`.
- Durante la verificación de funcionamiento de nginx, notamos que había un conflicto de puertos ocupados. Esto debido a que nginx intentaba ocupar el puerto 80, mientras que este estaba siendo ocupado por apache (que lo habíamos instalado unas clases atrás). Para resolverlo, detuvimos el servicio apache y deshabilitamos la inicialización automática.

Conclusiones y uso de IA

Nos pareció un proyecto muy interesante. El hecho de poder implementar un sistema de notificaciones nos abre un sinfín de usos que pueden relacionarse con domótica, monitoreo y automatización de tareas. Además, el haberlo realizado de forma local nos garantiza privacidad de datos y un control absoluto sobre el flujo de información.

En cuanto al uso de IA, la utilizamos para que nos ayude con diagnóstico de servicios, configuraciones de nginx, y auditoría de red. Nos recomendó utilizar comandos como `sudo /usr/bin/ntfy serve` (para ejecutar manualmente el servicio y ver los errores) a partir del prompt “El status de ntfy figura inactivo. Necesito saber el error específico. Que comando puedo utilizar en ubuntu?”, `sudo ln -sf /etc/nginx/sites-available/ntfy /etc/nginx/sites-enabled/` (para crear un enlace entre ambos directorios) y `sudo ss -tulpn | grep :80` (para visualizar quién estaba ocupando el puerto 80). Esto lo hicimos con el modelo *Gemini 3.1 Flash Lite*.