

Proyecto SyncBetter: Documentación de la configuración

Proyecto SyncBetter

Marian Georgian Ion

ÍNDICE

Introducción.....	2
Gestión de almacenamiento.....	2
Sistema operativo.....	4
Estructura de archivos y directorios.....	4
Servicios Principales.....	5
Otros Servicios.....	8
DNS (Local y Público).....	9
Red y Firewall.....	11
Scripts y Automatizaciones.....	12
Certificados SSL/TLS.....	12
Conclusiones.....	13

Introducción

Este documento describe la arquitectura y configuración de un servidor **autoalojado** con Ubuntu 22.04 LTS que corre múltiples servicios críticos (Nextcloud, Forgejo, Bitwarden, entre otros). El servidor base es un **HP Pavilion Elite HPE** al que se le han realizado mejoras de hardware, incluyendo la sustitución del procesador **Intel i5 750** por un **Intel i7 870** y la ampliación de la memoria RAM de **8 GB a 16 GB DDR3**. Estas actualizaciones permiten un mejor rendimiento y la capacidad de manejar múltiples servicios en simultáneo.

Uno de los principales objetivos es aprovechar al máximo los componentes de un ordenador más antiguo y convertirlo en un servidor que ofrece varios servicios personales.

La intención de este documento es guiar a lectores y profesionales interesados por los **detalles técnicos** de esta infraestructura, así como por las **buenas prácticas y lecciones aprendidas** durante el proceso de configuración. A lo largo de la documentación se describirán los **servicios alojados**, la **configuración de red y DNS**, las **normas de seguridad y firewall**, la **automatización de tareas**, y otros aspectos relevantes.

Este proyecto me ha llevado un año completo, desarrollando poco a poco cada una de sus características.

Gestión de almacenamiento

Uno de los aspectos más interesantes y complejos de este servidor es su **sistema de almacenamiento**, que combina varias unidades y particiones en diferentes configuraciones de **RAID1** y **LVM** para maximizar la capacidad, mantener la redundancia de datos y asegurar la disponibilidad de los servicios. A continuación se hace un breve resumen de la estrategia seguida:

Discos y particiones

- Se aprovechan **discos de diferentes tamaños** y tipos (HDD, particiones en un disco de 2 TB, etc.) para crear un entorno de almacenamiento flexible.
- Se emplea **RAID1** en la capa subyacente, pero con particiones específicas de varios discos físicos, de modo que el sistema se beneficia de la redundancia sin requerir que cada disco sea idéntico.

Gestión lógica con LVM

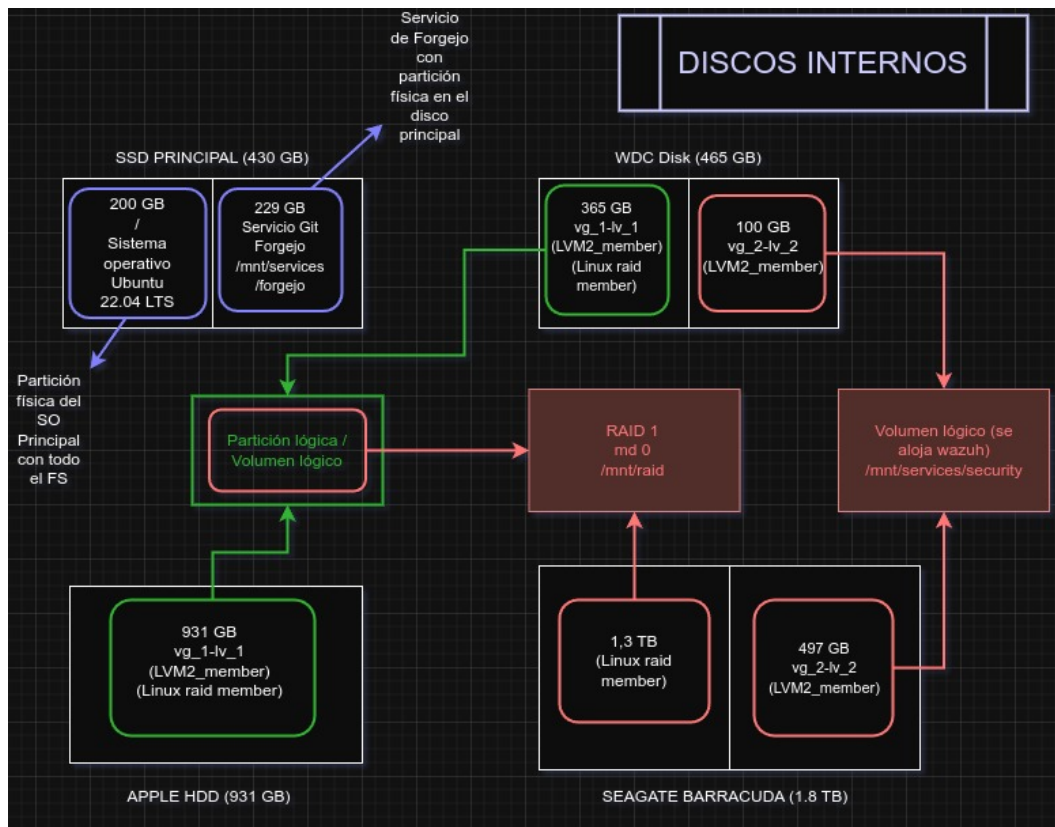
- Sobre la capa de RAID, se añade la **gestión de volúmenes lógicos (LVM)** para facilitar la ampliación, reducción y reorganización de los volúmenes de datos.
- Esta aproximación ayuda a administrar mejor la capacidad, permitiendo crear puntos de montaje o volúmenes dedicados para cada servicio (Nextcloud, Forgejo, etc.).

Discos externos para backup

- Se integran **discos externos** con distintos fines (backup, almacenamiento adicional), manteniendo copias de seguridad de los datos más críticos fuera del servidor principal.
- Esto aporta una capa adicional de protección ante fallos catastróficos y facilita la rotación de backups.

En total existen 7 discos añadidos al servidor, 4 internos y 3 externos, además de que uno de los externos dedicados a copias se enciende cada mes para recoger todas las copias proveniente de otro disco, por lo que no siempre está montado.

Los discos del servidor se estructuran de la siguiente forma.



Los discos externos se estructuran de la siguiente forma:



Sistema operativo

El sistema operativo es Ubuntu 22.04.5 LTS (Codename: jammy) y los usuarios son los que se observan en la imagen:

```
root@homeserver:~# cat /etc/passwd | grep "sh$"
root:x:0:0:root:/root:/bin/bash
marian:x:1000:1000:marian:/home/marian:/usr/bin/bash
ncadmin:x:1001:1001:::/home/ncadmin:/bin/bash
bitwarden:x:1002:1003::/opt/bitwarden:/bin/bash
git:x:120:129:Git Version Control,,:/opt/git:/bin/bash
root@homeserver:~#
```

Yo suelo operar casi siempre como “root” porque es el usuario más privilegiado para poder administrar estos servicios. El sistema operativo sigue unas políticas de actualizaciones correcta. Además, el administrador, mantiene actualizados los servicios, que siempre estén en su última versión. El administrador actualiza periódicamente el servicio de nube Nextcloud, Forgejo y Bitwarden, usando siempre que sea posible las últimas versiones de los servicios.

Ubicación

El servidor está ubicado en un entorno doméstico, y su objetivo principal es brindar de ciertos servicios a los usuarios que dispongan de las credenciales, como sincronización de datos con la nube personal, un repositorio privado además de GitHub donde alojar proyectos, y un gestor de contraseñas de código abierto y seguro como lo es Bitwarden, desplegado con contenedores usando Docker.

El acceso puede ser tanto interno como externo. Se realiza por el protocolo SSH y se usa una clave segura. Si el acceso es externo se realiza a través de una VPN configurada en el servidor y posteriormente por SSH.

El esquema de red no lo voy a compartir, ya que es muy sencillo. El servidor se conecta a un switch no gestionable en una habitación, el cual está conectado directamente al router principal ISP. Tiene una IP privada fija, 192.168.1.200/24 y tiene abiertos varios puertos con respecto a los servicios que ofrece. No voy a enseñar dichos puertos por seguridad pero existen varias reglas IPTABLES y UFW en el firewall de Linux que permiten el tráfico NAT a esos puertos, reglas de entrada y salida, y qué se realiza con ese tráfico antes de enrutarlo (PREROUTING) o después de enrutarlo (POSTROUTING).

Estructura de archivos y directorios

El sistema de archivos se organiza de esta forma:

- Nextcloud : “/var/www/nextcloud” para la web y “/mnt/raid/nextcloud” para volumen de datos nextcloud.
- Forgejo: /mnt/services/forgejo
- WAZUH: /mnt/services/security/wazuh
- Bitwarden: /opt/bitwarden/bwdata
- Copias de seguridad: “/mnt/backups”. Aquí dentro se estructuran según sean copias de archivos (files) o imágenes del sistema operativo (os_img).
- Disco externo que se añade a Nextcloud como “External Drive”: /mnt/external

Servicios Principales

Los principales servicios que manejo son:

Nextcloud: Nube personal donde gestiono todos mis archivos que se sincronizan desde el smartphone y portátil al servidor principal. Se realizó toda la instalación y configuración manual.

- Se usó Apache2 como servidor web, con varios archivos de configuración, varios módulos habilitados los cuales son los siguientes:

```
root@homeserver:/etc/apache2# ls mods-enabled/
access_compat.load  authz_user.load  filter.load  negotiation.conf  proxy_http.load  ssl.conf
alias.conf          autoindex.conf  headers.load  negotiation.load  proxy_wstunnel.load  ssl.load
alias.load          autoindex.load  http2.conf   php8.3.conf       remoteip.load      status.conf
auth_basic.load     deflate.conf     http2.load   php8.3.load       rewrite.load        status.load
authn_core.load     deflate.load     mime.conf    proxy.conf         sed.load            substitute.load
authn_file.load     dir.conf        mime.load    proxy.load         setenvif.conf
auth_core.load      dir.load        mpm_prefork.conf  proxy_connect.load  setenvif.load
authz_host.load     env.load        mpm_prefork.load  proxy_fcgi.load     socache_shmcb.load
```

- Se configuraron bastantes hosts virtuales, para cada uno de los servicios que ofrezco, tanto a nivel interno como de forma externa. He pixelado el nombre de algunos archivos por seguridad pero son varios, tanto a nivel interno como a nivel externo.

```
root@homeserver:/etc/apache2# ls sites-enabled/
00_apache_default.conf  06_git.syncbetter.com_80_443.conf
01_vault.syncbetter.com_80_443.conf  07_git.syncbetter.com_80_443.conf
03_cloud.syncbetter.com_80.conf      08_git.syncbetter.com_80_443.conf
04_cloud.syncbetter.com_443.conf
```

- La base de datos elegida des MariaDB ya que es comunitaria y de código abierto, además que permite algunos ajustes de forma gratuita que con MySQL hay que pagar el plan Enterorise Edition. Es completamente interna y sólo es accesible desde dentro del servidor. Tengo configuraciones de caché de memoria como Redis y APCu para el correcto uso de Nextcloud. En su documentación oficial hacen alusión a ello: Configurar Memory Caching Nextcloud.
- También ofrece autenticación F2A, que se realiza a través de una aplicación en el smartphone, en la cual se envía un código de 6 dígitos que posteriormente se coloca en un campo del formulario en Nextcloud. Esto se realiza a través del uso de llaves de seguridad físicas como Yubikeys.

Servidor de correo electrónico

Es importante configurar este servidor para que pueda enviar correos, por ejemplo para cambios de contraseñas y notificaciones.

Modo de envío: SMTP

Cifrado: SSL

Desde la dirección: cloud.syncbetter@gmail.com

Dirección del servidor: smtp.gmail.com : 465

Autenticación: ☒ Se necesita autenticación

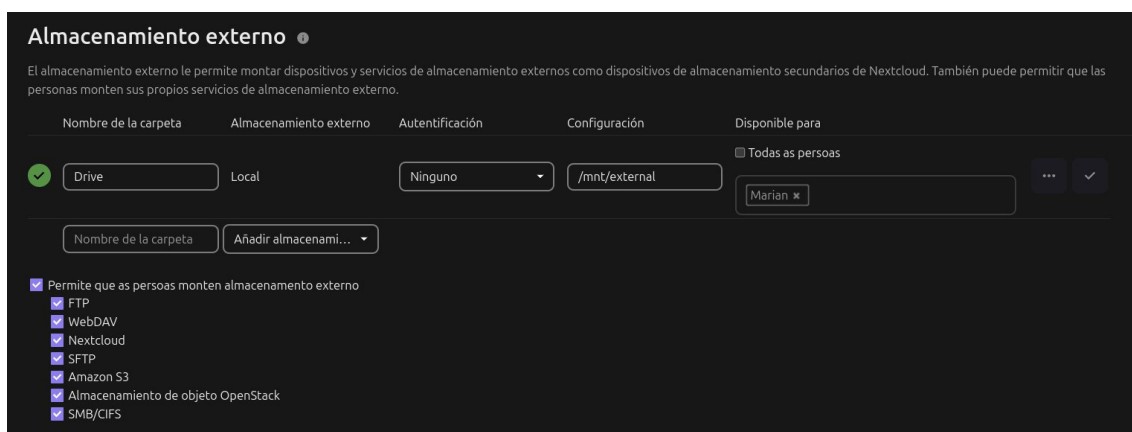
Credenciales: cloud.syncbett... [password masked]

Guardar

Comprobar y verificar configuración de correo: **Enviar mensaje**

- Tiene configurado un servidor de correo externo que permite enviar notificaciones de nextcloud, además de otras necesidades del propio servidor. Este correo se reutilizar para otros fines respecto al servidor, "cronjobs" ejecutados correctamente, etc.

- Existe una app de Nextcloud llamada “Suspicious Login” la cual permite registrar direcciones IP sospechosas y bloquearlas.
- También posee un antivirus para archivos, sobretodo cuando se sincronizan o descargan, llamado ClamAV.
- El disco externo mencionado anteriormente se configura desde Nextcloud como Almacenamiento externo, pero previamente debe estar montado. Esto lo realizo a través de scripts automatizados.



- También se usa otra app para fotos llamada “Memories” la cual es más eficiente que la app oficial. Otra app es la de “Recognize” que mediante modelos de Inteligencia Artificial que se deben descargar, etiquetan las fotos y vídeos de forma automática, para que su búsqueda sea más eficiente. GeoBloques es otra de la app que se debe tener para bloquear IPs procedentes de algunos países que no se deseen. Por último, también se ha configurado OnlyOffice Document Server en Nextcloud, para que se puedan editar archivos desde la propia nube, gracias a las aplicaciones necesarias para ello.

Forgejo: Servicio Git autoalojado para gestionar ciertos proyectos y código aparte de GitHub.

- Es excelente para subir código además de GitHub, donde se puede gestionar de otra forma diferente.
- Es un “fork” o forja de Gitea, y es comunitario y de código abierto. Permite gestionar repositorios y código de manera fácil y eficiente.
- Siguiendo la instalación desde esta página se configura y ya debería de funcionar: [Guía instalación Forgejo Linux](#).
- Cabe destacar que funciona a través del puerto 3000, pero si se requiere salir a internet a través de un subdominio se debe pasar a través de un proxy inverso como Nginx o Apache2. Como no tengo acceso al código fuente al instalar el binario, para poder unir Google Analytics con esta configuración se deben especificar ciertos parámetros de sustitución de cadenas en el virtual host, para que inserte el script necesario de Google Analytics para su correcto funcionamiento. También se ha configurado Google Tag Manager.


```

36 # Tiempos de espera del proxy
37 ProxyTimeout 720
38 ProxyPreserveHost On
39 ProxyPass / "http://127.0.0.1:3000/"
40 ProxyPassReverse / "http://127.0.0.1:3000/"
41
42 Header set X-Content-Type-Options "nosniff"
43 Header set X-XSS-Protection "1; mode=block"
44 Header set X-Frame-Options "DENY"
45
46 # Activa el filtro Substitute
47 AddOutputFilterByType SUBSTITUTE text/html
48
49 # Inserta el código de Google Analytics antes de la etiqueta </head>
50 Substitute "s#(<head[^\>]*>)#$1<!-- Google Tag Manager --><script>(function(w,d,s,l,i){w[l]=w[l]||[];w[l].push({'gtm.start': new
Date().getTime(),event:'gtm.js'}); var f=d.getElementsByTagName(s)[0], j=d.createElement(s),dl=l!='dataLayer'?'&l='+l:'';j.async
=true;j.src= 'https://www.googletagmanager.com/gtm.js?id=G-PLX50C1WSH';f.parentNode.insertBefore(j,f); })(window,document,'script','dat
aLayer','GTM-5GT2NHBS');</script><!-- End Google Tag Manager --><script async src='https://www.goog
letagmanager.com/gtag/js?id=G-PLX50C1WSH'></script><script>>window.dataLayer = window.dataLayer || []; function gtag(){dataLayer
.push(arguments);} gtag('js', new Date()); gtag('config', 'G-PLX50C1WSH');</script>#i"
51
52 Substitute "s#(<body[^\>]*>)#$1<!-- Google Tag Manager (noscript) --><noscript><iframe src='https://www.googletagmanager.com/ns.h
tml?id=GTM-5GT2NHBS' height='0' width='0' style='display:none;visibility:hidden'></iframe></noscript><!-- End Google Tag Manager
(noscript) -->#i"
53
54 LimitRequestBody 52428800

```

- Aún no tengo Forgejo funcionando con GitHub Actions o cualquier otra herramienta CI/CD aunque me gustaría probar.

Bitwarden: Gestor de contraseñas seguro y de código abierto, con capacidad de auto-alojar el servicio en Docker.

- Es uno de los mejores gestores de contraseñas de código abierto, y más seguros.
- Podemos observar, usando el comando de “docker ps”, aparecen todos los contenedores desplegados.
- En su [página de instalación](#) nos aparecen todos los pasos para poder configurar todo el entorno. Tiene ciertas variables de entorno modificadas para adaptarlas a mi entorno, al igual que el archivo “docekr-compose.yml”. Además existen ciertos directorios que se montan en los contenedores y los he modificado también, para que sólo me muestre los esencial.

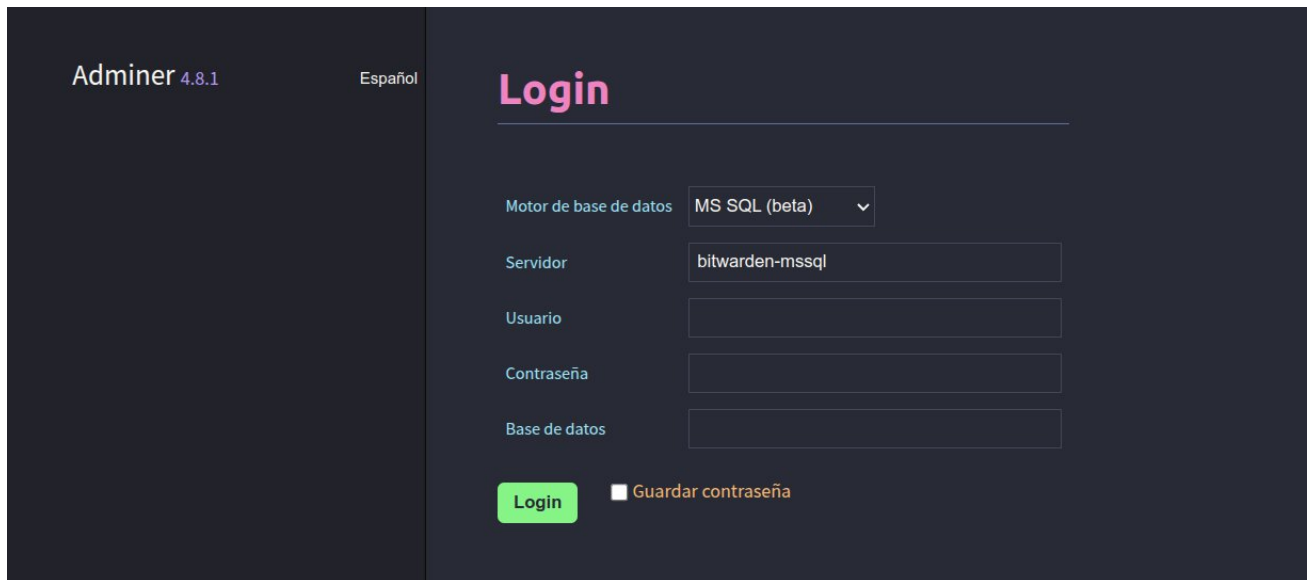
```
root@homeserver:/etc/apache2# docker ps | grep bitwarden | bat -l ruby
```

	STDIN
1	23de86ede909 bitwarden/nginx:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 80/tcp, 0.0.0.0:81->8080/tcp, [::]:81->8080/tcp, 0.0.0.0:444->8443/tcp, [::]:444->8443/tcp bitwarden-nginx
2	4e310a35e7d9 bitwarden/admin:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 5000/tcp bitwarden-admin
3	8f5c40cde72c bitwarden/identity:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 5000/tcp bitwarden-identity
4	89597f5f9dd3 bitwarden/attachments:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) bitwarden-attachments
5	c4cfe6752af6 bitwarden/events:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 5000/tcp bitwarden-events
6	891d332b2b4a bitwarden/notifications:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 5000/tcp bitwarden-notifications
7	e009980c2eb9 bitwarden/sso:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 5000/tcp bitwarden-sso
8	004b044aee3a bitwarden/mssql:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 1433/tcp bitwarden-mssql
9	797246a08fee bitwarden/api:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 5000/tcp bitwarden-api
10	e50f90ab93b0 bitwarden/web:2024.12.0 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) bitwarden-web
11	427d617dc357 bitwarden/icons:2024.12.1 "/entrypoint.sh" 3 months ago Up 29 hours (healthy) 5000/tcp bitwarden-icons

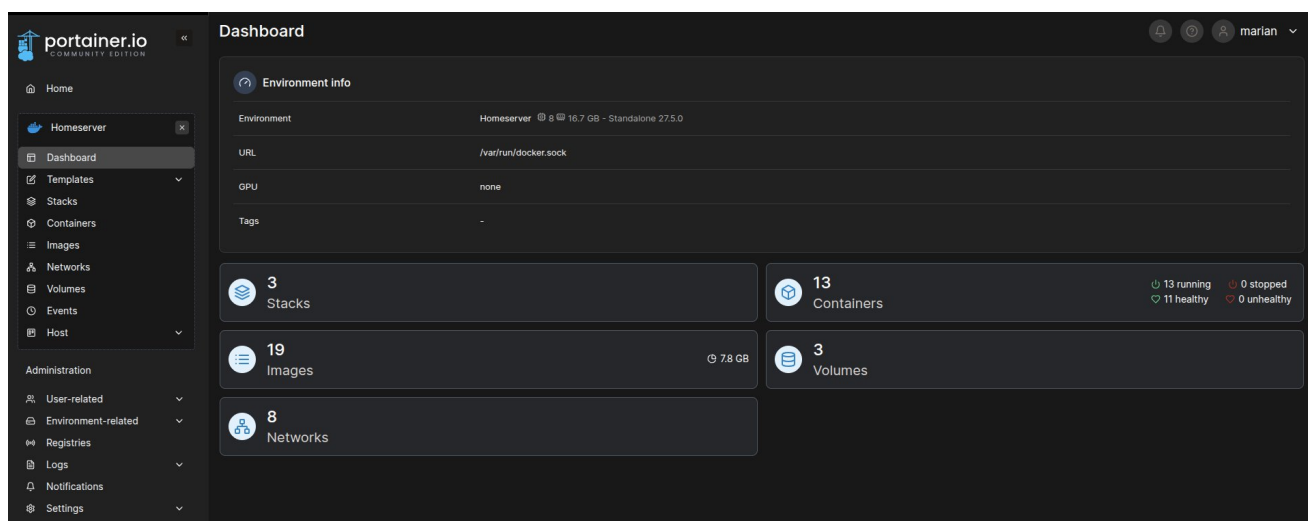
```
root@homeserver:/etc/apache2#
```

- También tiene configuradas las Yubikey, para poder acceder al gestor de forma segura. Se ha configurado además una cuenta de correo para este servicio, que permite enviar emails a los usuarios sobre la seguridad de las contraseñas, gestión de perfil, etc.

Adminer: Se ha configurado un servicio interno Adminer, que se conecta a las diferentes bases de datos y permite la fácil administración de las mismas, parecido a PhpMyAdmin.



Portainer: Se ha configurado un contenedor con Portainer, para facilitar el manejo y uso de contenedores Docker para el servidor.



Otros Servicios

Existen más servicios, como por ejemplo:

- Postfix: para permitir envío de email al administrador mediante comandos con “sendmail”. Esto sirve para las copias de seguridad.
- Redis y APCu para Nextcloud: son servicios aparte pero trabajan en conjunto con Nextcloud para gestionar la memoria caché.
- Wireguard: VPN funcional, la cual está configurada únicamente con los dispositivos que uso para entablar la conexión. Me sirve para acceder a servicios de forma interna y segura.
- Servidor DNS: funcional para la resolución de nombres a nivel interno para los diferentes servicios.
- Certificados SSL con Let's Encrypt para validar el dominio dinámico que posteriormente mencionaré.

DNS (Local y Público)

Este servidor dispone de DNS local y público. Vamos a explicar la configuración:

Para el DNS local se usa bind9. Mostraré los archivos de zona directa e inversa. El de zona directa vemos que resuelve a la misma IP todos los subdominios. Otra forma de haberlo realizado sería usar alias o CNAME para apuntar al servidor principal.

```
$TTL 86400
@      IN      SOA      homeserver.syncbetter.com.  root.syncbetter.com. (
        2025022501      ; serial
        21600           ; refresh after 6 hours
        3600            ; retry after 1 hour
        604800          ; expire after 1 week
        86400           ; minimum TTL of 1 day
)

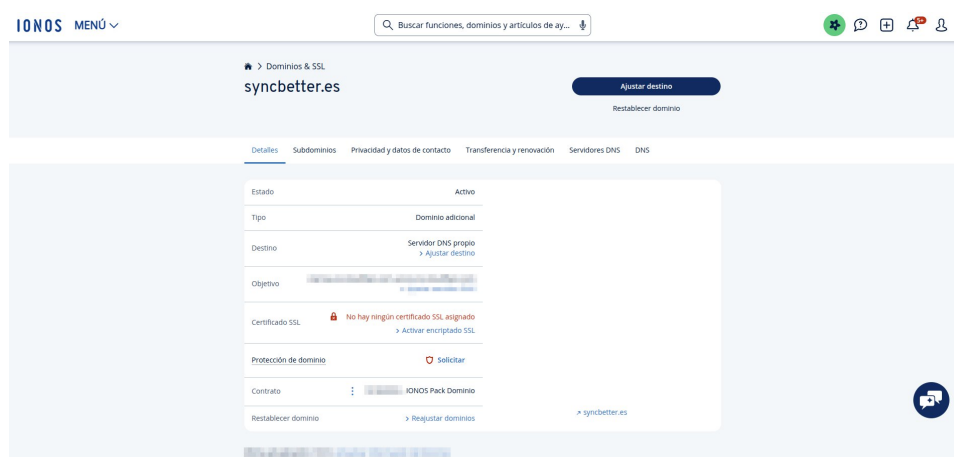
;
;      IN      NS       homeserver.syncbetter.com.
;
apiphp-grupo01 IN      A        192.168.1.200
homeserver      IN      A        192.168.1.200
portainer       IN      A        192.168.1.200
adminer         IN      A        192.168.1.200
proxy           IN      A        192.168.1.200
cloud           IN      A        192.168.1.200
vault           IN      A        192.168.1.200
git             IN      A        192.168.1.200
```

Para la zona inversa sólo se ha añadido el DNS del servidor que va a contestar las consultas. Es un DNS bastante sencillo para, se podría mejorar al instalar un contenedor en docker con la imagen Pi-Hole, que es una aplicación para bloqueo de anuncios y rastreadores en Internet a nivel de red en Linux que actúa como un sumidero de DNS, destinado para su uso en una red privada.

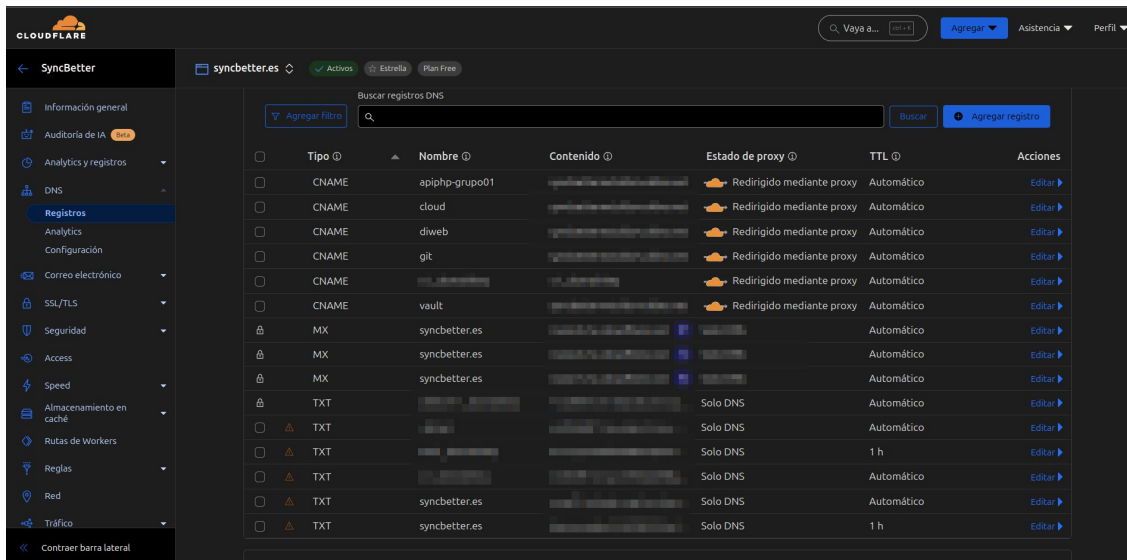
```
1 $TTL 86400
2 @      IN      SOA      homeserver.syncbetter.com.  root.syncbetter.com. (
3         2025022501      ; serial
4         21600           ; refresh after 6 hours
5         3600            ; retry after 1 hour
6         604800          ; expire after 1 week
7         86400           ; minimum TTL of 1 day
8     )
9 ;
10 ;      IN      NS       homeserver.syncbetter.com.
11 ;
12
13 200 IN      PTR       homeserver.syncbetter.com.
14
```

Para el DNS público se han realizado varios pasos:

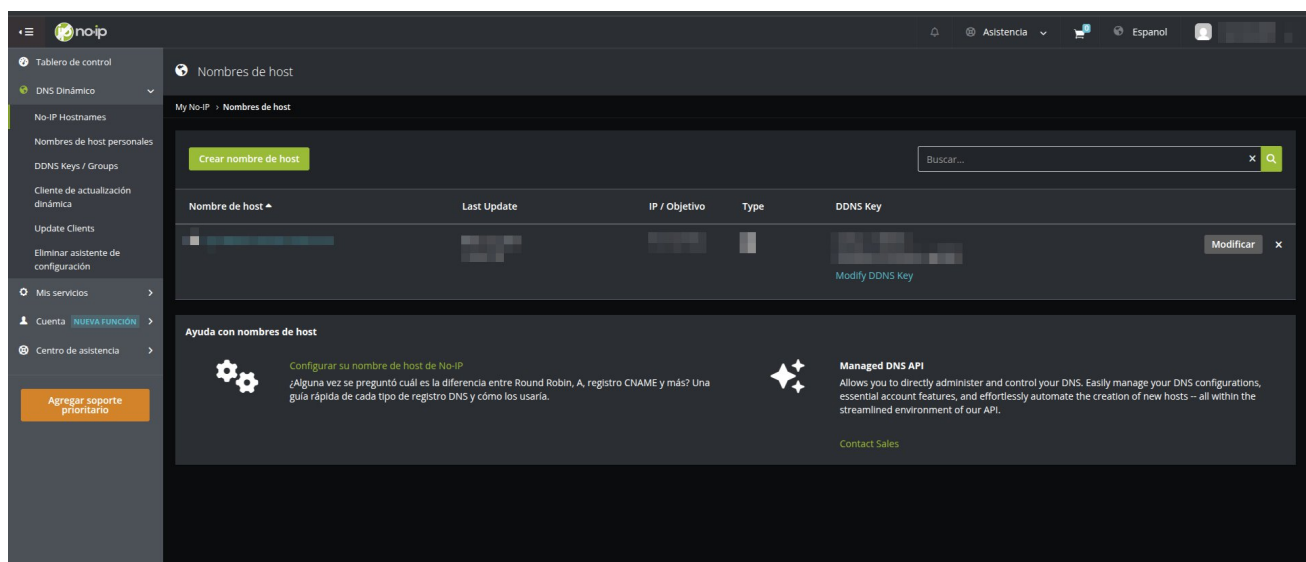
- Comprar dominio “syncbetter.es” con IONOS.



- Después se pasa todo el tráfico DNS a ese dominio por Cloudflare, para ofrecer la máxima protección posible.



- He pixelado la información por seguridad pero ahora mismo todo el tráfico hacia “syncbetter.es” pasa a través de Cloudflare. ¿Cómo redirijo desde Cloudflare a mi router personal? Muy fácil, empleando DNS Dinámico. Los DNS dinámicos o DynDNS se conocen al servicio capaz de asociar la IP pública en ese momento que se tiene al salir a internet con un nombre de dominio, pudiendo elegir el host.
- Ejemplo: yo uso No-IP como proveedor de DynDNS, y en su panel me permite elegir uno de los DNS disponible, por ejemplo “ddns.net”. Eso no se puede cambiar, pero sí se puede la parte del subdominio, es decir: “host.ddns.net”. En este momento “host.ddns.net” estará asociado a la IP pública que tenga en ese momento y realizará la traducción correctamente. Esto se debe a que no poseo una IP pública estática y mi proveedor no me permite esa opción, aunque no estoy en CG-NAT y me permite abrir puertos en el router.



- Finalmente, la petición llegaría a los puertos 80 o 443 de mi IP pública (gracias al DynDNS) y redirigirán al puerto e IP correspondientes del servidor en la red interna, el cual habrá un proxy como Apache2 escuchando dependiendo del servicio al que se dirija, por docker, por directorio como /var/www/nextcloud, etc.

Red y Firewall

Como se indicó anteriormente, se disponen de varias configuraciones a nivel de red y cortafuegos.

La topología de red es bastante sencilla, ya que el servidor conecta con un switch y posteriormente con el router ISP. En un futuro próximo me gustaría añadir un router aparte en la habitación del servidor y autogestionarlo, con políticas más restrictivas. Un buen router para esta administración y gestión podría ser MicroTik hAP ac², un modelo calidad-precio que ofrecería buena cobertura, además de incorporar RouterOS, desarrollado por MicroTik y muy gestionable para Administradores de redes y sistemas.

Se permiten ciertos puertos con ufw.

```
root@homeserver:~# ufw status | grep -v v6
Status: active

To Action From
--
22/tcp ALLOW Anywhere
80/tcp ALLOW Anywhere
443 ALLOW Anywhere
Bind9 ALLOW Anywhere
9 ALLOW Anywhere
3456 ALLOW Anywhere
81/tcp ALLOW Anywhere
444/tcp ALLOW Anywhere

root@homeserver:~#
```

Para la parte de IPTABLES, tengo configurados varios scripts que me permiten habilitar el acceso remoto o cerrarlo, hay varios puertos implicados ya que son intermediarios antes de llegar a los de destino final. Estos scripts sólo se lanzarían al indicar ciertos comandos de voz con el asistente Alexa. Tiene configurada una skill que permite ejecución de scripts usando mi cuenta de Amazon, para poder denegar el acceso si detecto que tengo un ataque. También podría conectarme por VPN y ejecutarlo manualmente.

```
root@homeserver:~/Scripts/Remote-admin# cat allow_public_access.sh deny_public_access.sh

File: allow_public_access.sh
1 #!/bin/bash
2
3 /usr/sbin/iptables -A INPUT -p tcp --dport 48443 -j ACCEPT
4 /usr/sbin/iptables -A INPUT -p tcp --dport 28443 -j ACCEPT
5 /usr/sbin/iptables -A INPUT -p tcp --dport 8443 -j ACCEPT
6 /usr/sbin/iptables -t nat -A PREROUTING -p tcp --dport 48443 -j REDIRECT --to-port 28443
7 /usr/sbin/netfilter-persistent save &>/dev/null

File: deny_public_access.sh
1 #!/bin/bash
2
3 /usr/sbin/iptables -t nat -D PREROUTING -p tcp --dport 48443 -j REDIRECT --to-port 28443
4 /usr/sbin/iptables -D INPUT -p tcp --dport 8443 -j ACCEPT
5 /usr/sbin/iptables -D INPUT -p tcp --dport 28443 -j ACCEPT
6 /usr/sbin/iptables -D INPUT -p tcp --dport 48443 -j ACCEPT
7 /usr/sbin/netfilter-persistent save &>/dev/null

root@homeserver:~/Scripts/Remote-admin#
```

El router tiene algunas reglas implementadas, para que todo lo que llegue por los puertos 80 y 443 lo redirijan al servidor, al igual que el puerto SSH correspondiente y el puerto UDP para VPN. Hay que recalcar que también se le ha configurado el DynDNS mencionado anteriormente para que el dominio se asocie a la IP.

Scripts y Automatizaciones

En mi cuenta de GitHub tengo un repositorio llamado “bash_utilities”, en el cual tengo varios scripts que uso en mi día a día. Esos scripts los usa también el servidor (no todos), sobretodo para gestión de copias de seguridad, apagado limpio, crontab, etc.

Enlace a mi repositorio: <https://github.com/mgion24/bash-utilities>

Se puede observar el “crontab” del usuario root, con las automatizaciones que éste realiza.

```
root@homeserver:~# crontab -l | awk 'NR>=25{print $0}'
# EMAIL DONDE RECIBIR LAS ALERTAS DE CRON
MAILTO=

# ESCRIBE '0' EN CADA ARRANQUE. NECESARIO PARA EL POSTERIOR APAGADO O REINICIO DEL SERVIDOR
@reboot echo "0" > /root/Scripts/Homesrv-admin/cron-management/is_executed

# MONTA EN CADA ARRANQUE LAS PARTICIONES DEL DISCO EXTERNO PARA MOSTRAR EN LA NUBE
@reboot /bin/bash /root/Scripts/Local-admin/mount_external_disk_parts.sh

# CRON NECESARIO PARA NEXTCLOUD
*/5 * * * * /bin/bash /root/Scripts/Homesrv-admin/cron-management/cron_nextcloud.sh

# MONTA LOS DISCOS DE COPIAS DONDE SE REALIZARAN LAS COPIAS SEMANALES
* * * * * /bin/bash /root/Scripts/Local-admin/mount_bak_disks.sh

# DESMONTA LOS DISCOS DE COPIAS (MEJOR DESMONTAR CUANDO ACABEN LAS COPIAS)
#32 02 * * 1 /bin/bash /root/Scripts/Local-admin/umount_bak_disks.sh

# MONTA EL DISCO DE COPIAS DE MAS ALMACENAMIENTO PARA PASAR TODO DE LOS OTROS DISCOS
# A ESTE, SIENDO EL MONTAJE UNA VEZ AL MES
#30 02 * * 1 /bin/bash /root/Scripts/Local-admin/mount_main_bkp_disk.sh

# DESMONTA EL DISCO DE COPIAS PRINCIPAL (MEJOR DESMONTAR CUANDO ACABEN LAS COPIAS)
#30 02 * * 1 /bin/bash /root/Scripts/Local-admin/umount_main_bkp_disk.sh

# CREACION DE COPIAS DE ARCHIVOS IMPORTANTES
#30 12 * * 6 /bin/bash /root/Scripts/Homesrv-admin/backups-management/create_backup.sh

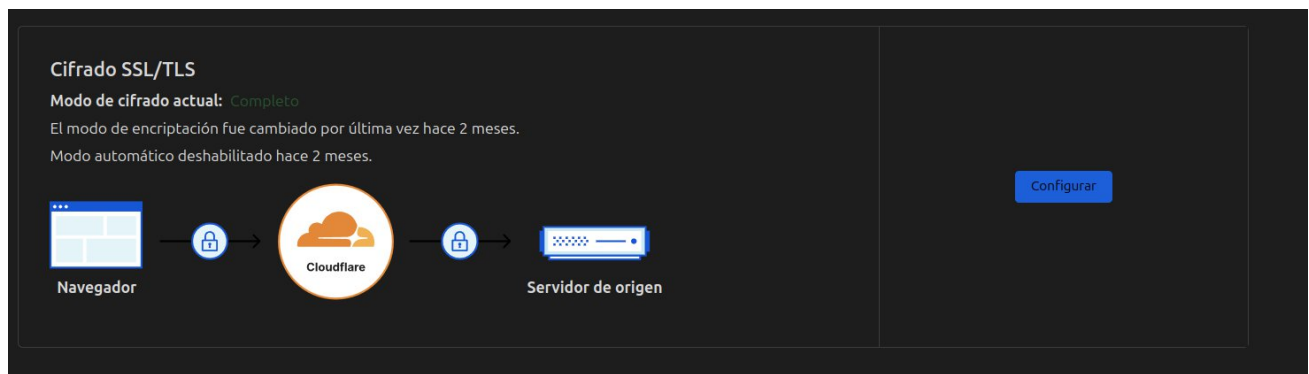
# CREACION DE COPIAS DE BITWARDEN
30 12 * * * /bin/bash /root/Scripts/Homesrv-admin/bitwarden-management/backup-db-mssql.sh

# PRUEBAS
# */3 * * * * /bin/bash /root/Scripts/Homesrv-admin/shutdown-management/shutdown.sh r
root@homeserver:~#
```

Las notificaciones llegan al email del administrador, ya sea por medio de crontab o usando “sendmail” en un script. Todavía no he usado Prometheus o Nagios para monitorización pero me gustaría probar esos servicios.

Certificados SSL/TLS

Ya mencioné en apartados anteriores que uso Let’s Encrypt para que el DynDNS esté validado con mi IP, pero realmente, con el dominio de “syncbetter.es”, se encarga Cloudflare asignarme un certificado para el sitio. En la siguiente imagen se observa que tengo el modo de cifrado completo, por parte de Cloudflare.



También posee todas las políticas DMARC, DKIM y SPF configuradas correctamente. Para ello se tuvieron que añadir ciertos registros TXT que validaban estas políticas, además de otros tipos de DNS como CNAME, etc. Habilité el DNSSEC, el cual asegura la integridad y autenticidad de mi dominio web. Para ello tuve que ponerme en contacto con IONOS para solicitar cierta información que me pedía Cloudflare.

Conclusiones

Antes de terminar, quiero mencionar que esta es una fase inicial de toda la documentación que proporcionaré para este proyecto.

Este proyecto lo he realizado estando en primer año de DAW, ya que recién había acabado ASIR y disponía de más tiempo para investigaciones. Me ha ayudado a profundizar mucho en la administración de sistemas, sobre todo en sistemas Linux, y a entender y arreglar errores que se podrían dar en un caso real de uso, aprendiendo a gestionarlos y buscar la mejor solución posible. Me ha permitido, además, aplicar y consolidar todos mis conocimientos que adquirí en ASIR para un proyecto que actualmente le sigo dando uso. Lo considero uno de los mejores proyectos que realicé, junto con el TFG de ASIR.

Toda la documentación se encuentra en [mi GitHub](#).