Instalación y configuración de alta disponibilidad de sistemas de backup

INDEX

INFRAESTRUCTURAS DE HARWARE:

- Servidor principal:

  HPE ProLiant DL380 Gen10.

  Especificaciones – CPU Intel Xeon escalable 4210 con 10 núcleos a 2.2 HZ

  Memoria RAM: 64GB DDR4 (4modulos de 16 GB)
  almacenamiento: 100 TB en configuración RAID 10 (ampliable)

portátil:

- Marca: HP
- *Sistema operativo: Debian 12 sin interfaz grafica*

JUSTIFICACIÓN:

- **La elección del servidor HPE ProLiant DL380 Gen 10 se basa en su velocidad y fiabilidad. Este servidor ofrece un rendimiento superior, integrando características avanzadas de gestión y seguridad. La memoria RAM de 64 GB permite el manejo eficiente de grandes conjuntos de datos y facilita la ejecución fluida de aplicaciones exigentes. Además, el almacenamiento RAID 10 de 100 TB proporciona una sólida protección de datos y garantiza una alta disponibilidad del sistema, minimizando el tiempo de inactividad.**

Servidor respaldo:

- Servidor en la nube (aws)Amazon Elástic compute cloud (Amazon EC2)

  CPU contratado 8 núcleos CPU

  4 memoria RAM 64GB

  Almacenamiento 100 TB.

Instancia M5.2 large

Justificación:

- Se ha elegido esta configuración debido a su alta escalabilidad y disponibilidad, lo que garantiza que el servidor de respaldo pueda crecer según las necesidades del hospital y esté siempre disponible cuando se necesite. Además, AWS ofrece una garantía de disponibilidad del 99,99%, lo que proporciona una capa adicional de confiabilidad y seguridad para los datos del hospital.

Sistema de redundancia y replicación Configuración (activo-pasivo) mecanismo asíncrono.

- Se ha establecido una configuración activo-pasivo, donde el servidor físico actúa como servidor principal y el servidor en la nube funciona como servidor secundario. La replicación de datos se realiza de manera asíncrona, lo que significa que la sincronización entre los servidores ocurre eventualmente, por eventos.

Justificación:

- Esta configuración ofrece una combinación de alta disponibilidad, tolerancia a fallos y capacidad de respuesta del sistema, lo que la hace ideal para respaldar las necesidades críticas de datos del hospital. Aunque tiene sus contras, como costos adicionales y complejidad de configuración, pero los beneficios en términos de seguridad y continuidad del servicio superan ampliamente estas consideraciones

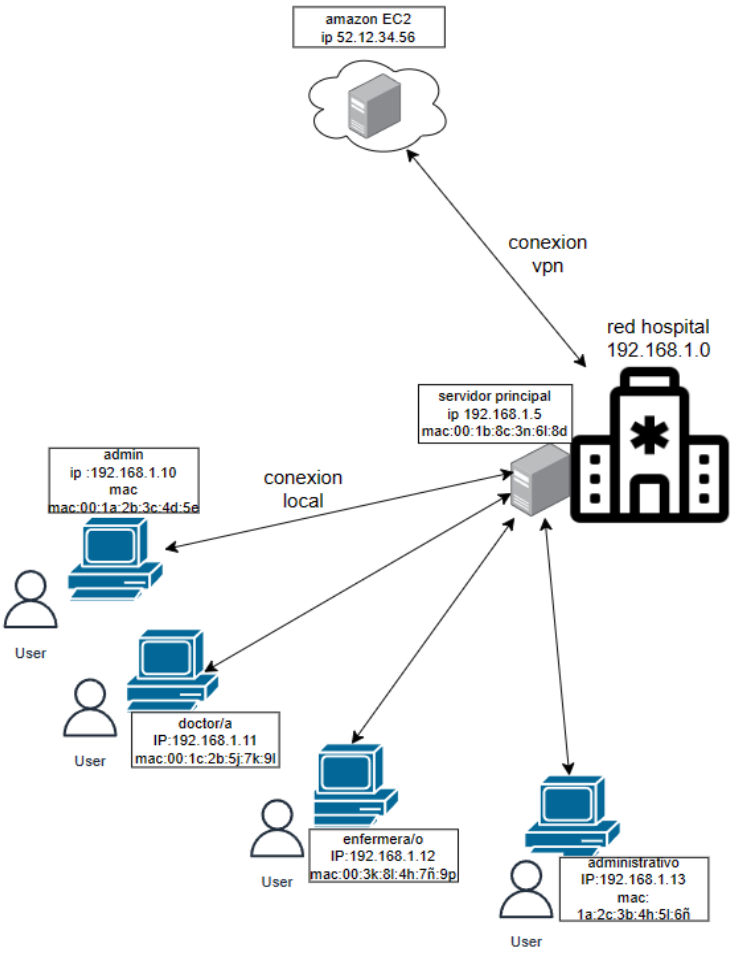
Configuración activo-pasivo:

- En esta configuración, el servidor principal se encuentra en el hospital de Blanes y actúa como el nodo activo, encargado de todas las operaciones de lectura y escritura. Este nodo es el punto central de la infraestructura y maneja todas las tareas críticas para el funcionamiento del sistema.

- Por otro lado, el segundo nodo se encuentra en la nube (**cloud**) y opera como el nodo pasivo. En este caso, el servidor en la nube permanece inactivo la mayor parte del tiempo, solo activándose en caso de que el nodo principal en el

hospital falle o experimente algún problema. En tal situación, el servidor en la nube asume automáticamente las funciones del servidor principal para garantizar la continuidad del servicio y minimizar cualquier impacto en las operaciones del hospital.

Diagrama de funcionamiento:



Manual de instalación y como se administran de los servidores

1. Instalación del Servidor Físico en el hospital:

- Determinas el hardware y sistema operativo del servidor físico. En nuestro caso es Debian12.
- Configuración de la red del servidor para que esté conectada a la misma red que los usuarios del hospital.
- Instalar Software necesario para llevar la gestión del hospital.

2. Configurar el servidor de Réplica en la Nube en Amazon

- Crear la cuenta de AWS
- Lanzar una instancia EC2
- Configuración de la seguridad. Hay que destacar que es muy importante tener unos estándares de seguridad altos ya que tratamos con datos personales de pacientes.
- Instalación del software necesario en la instancia EC2. Herramientas de monitorización, software de respaldo, etc.
- Configuración de la Replicación de Datos. Los datos al replicarse en ambos servidores tienen que estar disponibles por si uno de ellos cae.

3. Administrar ambos servidores

- Monitorizar ambos servidores

- Gestionar de usuarios y permisos

- Configuración de las copias de Seguridad

- Configurar la seguridad en ambas maquinas. Como firewalls, VPN, detección de intrusiones.

4. Técnica de balanceo

- Podemos utilizar la técnica “Route 53 Latency Based Rotuing” que proporciona AWS. Esta técnica aprovecha la red para dirigir el tráfico de los usuarios al servidor que tenga menos latencia.

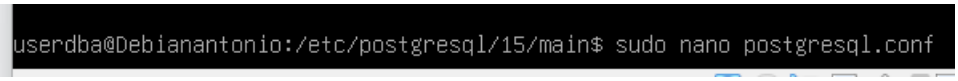
Configuración para tener alta disponibilidad de sistema de backup

- Primero creamos la máquina virtual y configuramos el postgres
- Una vez creada, la replicamos en otra MV para poder tener alta disponibilidad.

Pasos para tener alta disponibilidad:

En el fichero de postgresql.conf descomentamos las siguientes líneas:

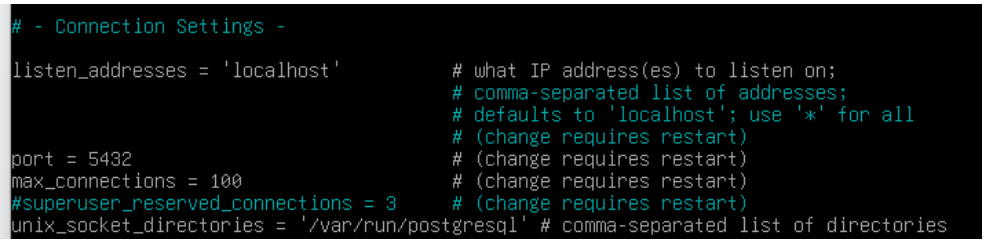
Abrimos el fichero



Habilitar la replicación WAL: Aseguramos que el servidor principal esté escribiendo los registros WAL.

Configuramos el método de envió. Sincrónicamente se enviarán los registros WAL.

Especificamos la IP del servidor principal. En nuestro es localhost, que es la IP 192.168.1.137.



Configuramos la identificación de la replicación proporcionando las credenciales del segundo servidor.

Habitamos los registros de replicación.

Abrimos el fichero pg_hba.conf. Este archivo controla las conexiones a la base de datos. Añadimos la línea al final del fichero.

```
host    replication     userdba         192.168.1.137/32        md5
```

Aplicamos los cambios reiniciando el servicio

```
userdba@Debianantonio:/etc/postgresql/15/main$ sudo systemctl restart postgresql
userdba@Debianantonio:/etc/postgresql/15/main$ sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
     Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; preset: enabled)
     Active: active (exited) since Sun 2024-04-28 19:54:59 CEST; 10s ago
    Process: 1532 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
   Main PID: 1532 (code=exited, status=0/SUCCESS)
        CPU: 3ms

Apr 28 19:54:59 Debianantonio systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Apr 28 19:54:59 Debianantonio systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
```

Entramos en la base de datos y creamos el usuario de replicación.

```
userdba@Debianantonio:/etc/postgresql/15/main$ sudo -u postgres psql
perl: warning: Setting locale failed.
perl: warning: Please check that your locale settings:
        LANGUAGE = (unset),
        LC_ALL = "es_ES:UTF-8",
        LANG = (unset)
    are supported and installed on your system.
perl: warning: Falling back to the standard locale ("C").
psql (15.3 (Debian 15.3-0+deb12u1))
Type "help" for help.

postgres=# create user userdba with password '123456';
CREATE ROLE
postgres=# alter user userdba superuser;
ALTER ROLE
postgres=#
```

Ahora vamos al servidor secundario, nos tenemos que asegurar que este en la misma red que el primario y tenga postgresql.

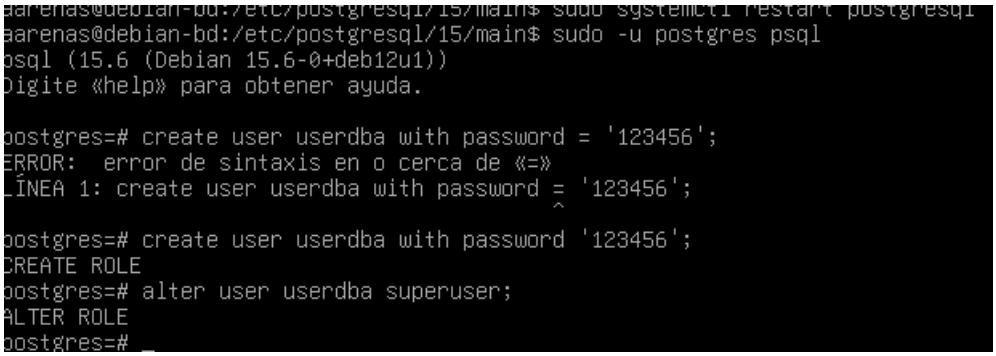
- Seguimos los mismos pasos que el primario
- En esta máquina hay que poner la dirección del servidor principal en el fichero postgresql.conf.

```
primary_conninfo = 'host=192.168.1.137 port=5432 user=userdba password=123456'
```

Configuramos el fichero pg_hba.conf para indicarle usuario de replica.

```
hostssl all             all             0.0.0.0/0               md5
host    replication     userdba         192.168.1.136/32        md5
```

Creamos el usuario de replica



Establecimiento de conexión vpn

- Primero se configuran los parámetros de replicación en el nodo principal, incluida la habilitación del registro binario (binlog). Luego, se configura el nodo secundario para que se suscriba al binlog del nodo principal y comience a replicar los datos.

Configurar pruebas Failover

- Este modo operativo de respaldo, o Failover es para evitar que no haya perdida de datos, y garantizar la continuidad del servicio en caso de que el componente principal deje de estar disponible ya sea debido a una falla imprevista o a un tiempo de inactividad programada decir, si el componente principal deja de estar disponible, ya sea por falla o por tiempo de inactividad programada.

INSTALACION Y CONFIGURACION BACKUPS:

Instalamos cron una herramienta la cual nos permite crear tareas y que se ejecuten de manera automática.

1. Sudo apt update
2. Sudo apt install cron

Para poder acceder al fichero donde se crearán las tareas:

- Crontab -e

Creación y configuración del script de backup:

```
#variables
PG_USER="postgres"
PG_PASSWORD="Blanes2121"
PG_HOST="localhost"
BACKUP_PATH="/etc/postgresql/15/main/backups/backups_logicos"
LOG_FILE="/etc/postgresql/15/main/logs_backups/backup.log"

# funcion backup logico
backup_logico() {
    DB="hospital"
    FILE="$BACKUP_PATH/$DB-$(date +%Y%m%d).sql"
    PGPASSWORD="${PG_PASSWORD}" pg_dump -h "${PG_HOST}" -U "${PG_USER}" "${DB}" > "$FILE"
    gzip $FILE
    if [ $? -eq 0 ]; then
        echo "$(date +"%Y-%m-%d %H:%M:%S") - Backup lógico de $DB completado correctamente" >> "$LOG_FILE"

    else
        echo "$(date +"%Y-%m-%d %H:%M:%S") - ERROR: Backup lógico de $DB fallido" >> "$LOG_FILE"
    fi
}
```

En le apartado **variables** se configuraron datos necesarios para que el script pueda acceder a la base de datos para asi buscar y crear la copia de seguridad.

La función backup logico genera copia de la base de datos especificada (DB) la cual es la principal base de datos una vez obtenida la comprime y en caso de producirse un error se crearan logs.

Notificaciones de backup.

```
# Función para correo electrónico
mensaje_gmail() {
curl --url 'smtps://smtp.gmail.com:465' \
  --ssl-reqd \
  --user "a.arenas@sapalomera.cat:F1O9X8Y7" --insecure \
  --mail-from 'a.arenas@sapalomera.cat' \
  --mail-rcpt 'a.arenas@sapalomera.cat' \
  --upload-file "/etc/postgresql/15/main/aviso_logico.txt"
}

# Función para bot de Telegram
mensaje_bot_telegram() {
    TOKEN="7022652300:AAFm0A3m1hO7xgUACKIfbvzZF3kD33fC9CI"
    CHATID="6103202840"
    URL="https://api.telegram.org/bot${TOKEN}/sendMessage"
    DATE=$(date +"%d-%b-%Y")
    TEXT="Còpia de seguretat realitzada ${DATE}"

    curl -d "chat_id=${CHATID}&disable_web_page_preview=1&text=${TEXT}" "$URL"
}
```

Para añadir el script tiene implementado unas funciones que notificaran al usuario mediante Gmail y telegram.

Configuración del cron:

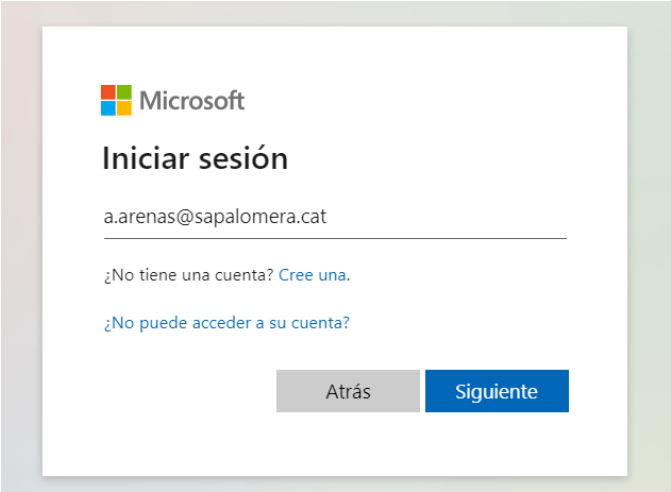
```
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
# 0 0 * * 1-5 sh /etc/postgresql/15/main/scrip_backup_logic.sh
# 50 20 * * 1-5 sh /etc/postgresql/15/main/scrip_backup_fisic.sh
  0 23 * * 1-5 sh /etc/postgresql/15/main/script_backup.BD.sh
```

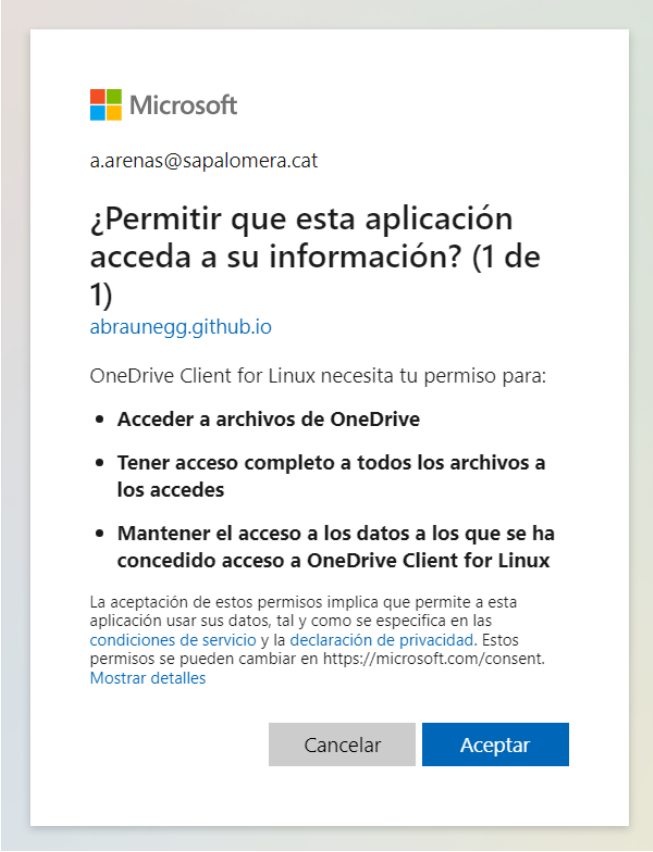
Guardar en la nube:

Para instalar onedrive en debian se aplicaron las siguientes comandas:

sudo -- bash -c 'apt update && apt install --yes onedrive'

una vez instalado escribimos por terminal onedrive la primera vez nos mostrara un link el cual tenemos que copiar y poner en el navegador, una vez en este tendremos que iniciar sesión y darle permiso para que acceda a la información.





Una vez terminamos de hacer lo anterior copiamos la url y la ponemos en el terminal, no retornara que la autorización ha sido un éxito.

```
aarenas@debian-bd:/etc/postgresql/15/main$ onedrive

WARNING: Your onedrive client version is now obsolete and unsupported. Please upgrade your cl
ient version.
Current Application Version: 2.4.23
Version Available:           2.4.25

Configuring Global Azure AD Endpoints
Authorize this app visiting:

https://login.microsoftonline.com/common/oauth2/v2.0/authorize?client_id=d50ca740-c83f-4d1b-b
616-12c519384f0c&scope=Files.ReadWrite%20Files.ReadWrite.All%20Sites.ReadWrite.All%20offline_
access&response_type=code&prompt=login&redirect_uri=https://login.microsoftonline.com/common/
oauth2/nativeclient

Enter the response uri: https://login.microsoftonline.com/common/oauth2/nativeclient?code=M.C
545_BL2.2.U.b9446b19-996c-50a2-9ce5-bf4ee8b501f0

Application has been successfully authorised, however no additional command switches were pro
vided.

Please use 'onedrive --help' for further assistance in regards to running this application.

aarenas@debian-bd:/etc/postgresql/15/main$
```

Ahora lo sincronizamos:

```
aarenas@debian-bd:~$ onedrive --synchronize

WARNING: Your onedrive client version is now obsolete and unsupported. Please upgrade your cl
ient version.
Current Application Version: 2.4.23
Version Available:           2.4.25

Configuring Global Azure AD Endpoints
Initializing the Synchronization Engine ...
Syncing changes and items from OneDrive ...
Downloading file ./Introducción a OneDrive.pdf ... done.
Creating local directory: ./Documentos
Creating local directory: ./Imágenes
Performing a database consistency and integrity check on locally stored data ...
Uploading differences of ~/OneDrive
Uploading new items of ~/OneDrive
Sync with OneDrive is complete
aarenas@debian-bd:~$
```

Una vez terminado ahora tendremos creada una carpeta onedrive, para ver la ruta podemos hacer **onedrive --display-config,** en esa ruta se creara una carpeta para los backups.

```
aarenas@debian-bd:~$ onedrive --display-config
onedrive version                             = v2.4.23-1
Config path                                  = /home/aarenas/.config/onedrive
Config file found in config path             = false
Config option 'sync_dir'                     = /home/aarenas/OneDrive
```

Se añadieron dos líneas mas, una para hacer una copia del comprimido y después una para sincronizar el onedrive:

```
backup_logico() {
    DB="hospital"
    FILE="$BACKUP_PATH/$DB-$(date +%Y%m%d).sql"
    PGPASSWORD="${PG_PASSWORD}" pg_dump -h "${PG_HOST}" -U "${PG_USER}" "${DB}" > "$FILE"
    gzip $FILE
    cp "$FILE.gz" "/home/aarenas/OneDrive/backups/"
    if [ $? -eq 0 ]; then
        echo "$(date +"%Y-%m-%d %H:%M:%S") - Backup lógico de $DB completado correctamente" >> "$LOG_FILE"
    else
        echo "$(date +"%Y-%m-%d %H:%M:%S") - ERROR: Backup lógico de $DB fallido" >> "$LOG_FILE"
    fi
}


sincronizar() {
onedrive --synchronize --upload-only
}
```

Como podemos ver el fichero se subió correctamente