Proyecto Final de la Muerte

EVIL CORP

Ian Nogueira
30/04/2023
Assegurament De La Informació

1.INDICE

1.INDICE_____ **2**

2.DIAGRAMA_____**3**

3.SQL_____**5**

CREACIÓN BASE DE DATOS Y TABLA_____ 5

CREACIÓN DE USUARIO “HILON MUSGO”_____ 6

USUARIOS READER Y EDITOR_____ 6

COMPROBACIONES_____ 7

4.HARDWARE_____ **11**

REQUERIMIENTOS_____ 11

COMPONENTES DEL HARDWARE_____13

CONFIGURACIÓN DISCOS DUROS_____ 15

PRESUPUESTO DE LA BASE DE DATOS_____ 16

5.CLUSTER_____ **17**

DISEÑO CLUSTER_____ 17

6.COPIAS DE SEGURIDAD_____ **18**

¿Qué hardware compraréis para ayudaros?_____ 18

¿Cómo y cuándo se realizan las copias de seguridad?_____19

¿Qué ocurre si falla un disco duro y cómo debemos recuperarlo? ¿Cómo entra en juego la configuración de los discos duros?_____ 19

¿Y si falla una máquina entera?_____ 19

7.CRON_____ **20**

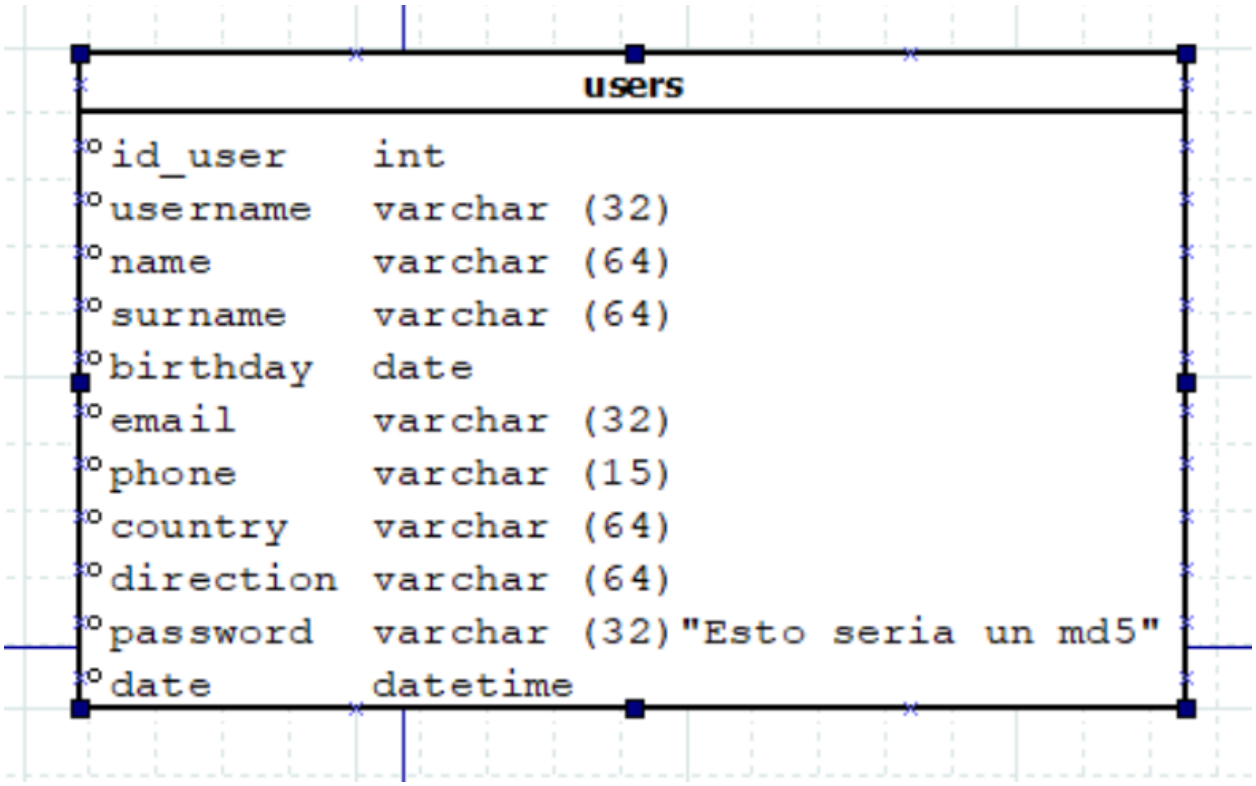
SCRIPT DE LAS COPIAS DE SEGURIDAD_____20

LINEAS DEL CRONTAB_____ 23

8.COMPRESION_____ **24**

2.DIAGRAMA

Primeramente, a la hora de hacer nuestro diagrama de la tabla de usuarios tendremos que pensar en qué datos necesitamos. Los datos que una corporación de Hilon Musgo necesita suelen ser muy abundantes y personales, pero como no queremos sobrecargar la base de datos con tanta información lo que haremos será reducir esos datos a los más esenciales. Que son estos:



Id_user: *EL id es un campo necesario para poder gestionar cualquier base de datos, este campo estaría oculto al cliente.*
Username: *Este campo lo utilizaría Hilon Musgo para ser root y para el resto de clientes este será el nombre que verán el resto de usuarios.*

Name: *El nombre es necesario para poder identificar a la persona y contiene 64 caracteres porque el nombre más largo del mundo registrado ocupa 39 caracteres.*

Surname: *Los apellidos del cliente son necesarios para poder identificar a la persona. Tiene un máximo de 64 caracteres porque serán los dos apellidos y el apellido más largo del mundo contiene 36 caracteres.*

Birthday: *La fecha de nacimiento es importante para saber cuantos años tiene la persona en concreto y para saber si es mayor de edad o no.*

Phone: *El número de teléfono es importante ya que es una buena forma de contactar a la persona en concreto y una manera de enviar publicidad. El campo ocupa 15 espacios ya que cuenta con que se le añadirá el prefijo del país correspondiente y los espacios necesarios para separar el número de teléfono.*

Country: *El país es necesario para saber localizar a la persona en concreto y poder hacer un estudio de mercado más profundo y personalizado. Tiene 64 caracteres por que el país con más caracteres contiene 50.*

Dirección: *La dirección es importante ya que se puede determinar el nivel adquisitivo de la persona para luego hacer un estudio de mercado más personalizado. Contiene 64 caracteres ya que el nombre más largo del mundo es de 29 caracteres sin contar número, puerta y piso.*

Contraseña: *La contraseña para poder acceder a los datos. Es de 32 caracteres ya que cuando la introduzcan se transformará inmediatamente en un MD5 que ocupa 32 caracteres.*

Fecha de inscripción: *La fecha de inscripción es importante para mantener un registro de cuando entra un cliente o para saber si las acciones publicitarias concretas funcionan.*

3.SQL

CREACIÓN BASE DE DATOS Y TABLA

A la hora de crear nuestro script donde se creará toda la base de datos y la tabla primero tenemos que saber los campos de la tabla. Como ya los tenemos gracias al diagram solo hará falta crear dicha sentencia, que quedaria algo asi:

```
1    create database if not exists EvilCorp;
2    drop table if exists `users`;
3    create table `users`(id_user BIGINT UNSIGNED NOT NULL
AUTO_INCREMENT PRIMARY      KEY, username VARCHAR(32) NOT NULL,
name VARCHAR(64) NOT NULL, surname VARC     HAR(64), birthday
DATE NOT NULL, email VARCHAR(32) NOT NULL, phone VARCHAR(1    5)
NOT NULL, country VARCHAR(64) NOT NULL, direction VARCHAR(64),
password V    ARCHAR(32), date DATETIME NOT NULL DEFAULT now());
```

Con estas sentencias lo que haremos primero será crear la base de datos de EvilCorp si no existe ninguna, si ya existe alguna base de datos con ese nombre no creará ninguna. Luego eliminará la tabla de users si existe alguna con ese nombre. Después de borrar la tabla, creara dicha tabla users con todos los parametros y campos mencionados anteriormente. La única cosa a destacar en la parte de la creación de la tabla es en la parte del campo DATE, este campo se establece que por defecto será justo la hora exacta que se registre algún usuario.

CREACIÓN DE USUARIO “HILON MUSGO”

La segunda parte del script que haremos será para añadir a nuestro fundador Hilon Musgo como el primer usuario en esta base de datos. Hilon Musgo será root de la base de datos. Para añadir a Hilon Musgo añadiremos esta línea al script:

```
4    insert into `users`(username, name, surname, birthday, email, phone, country     , direction, password) values ("root", "Hilon",          "Musgo",          "1971-06-28",          "hilonmusgo@gmail.com","27666666666","South Africa","EvilCorp666","e1e71757deb074    60abff6678e3cd468f");
```

Esta línea lo que hará será insertar en la tabla de usuarios al señor Hilon Musgo con el usuario root. Esto hará que Hilon Musgo sea el usuario con id 1 y con toda la información correspondiente. Las únicas partes a destacar son que no se inserta el campo de datos y que la password es algo extraña. El primer caso es por que como hemos puesto por defecto que sea la hora actual, no hace falta añadirla luego al insert ya que se pondrá sola. El segundo caso es por que simula que es una contraseña en MD5. La contraseña es EvilCorpMagnate pero está encriptada en MD5.

USUARIOS READER Y EDITOR

La tercera y última parte del script para esta base de datos será la creación de los usuarios que serán capaces de leer la tabla de usuarios, y el que podrá editar dicha tabla. La línea para añadir a estos dos usuarios será esta:

```
    create user 'editor'@'localhost' identified by 'enti';
```

```
    create user 'reader'@'localhost' identified by 'enti';
    grant insert, update on EvilCorp.* to 'editor'@'localhost';
    grant select on EvilCorp.* to 'reader'@'localhost';
    flush privileges;
```

Con estas líneas podremos crear el usuario editor en el host local, y el reader, que también estará en el host local. Luego de crear estos dos usuarios, les daremos permisos para que primero, el editor sea capaz de hacer inserts y updates y segundo, el usuario reader sea capaz de hacer selects en la base de datos de EvilCorp. Estos usuarios solo tienen permisos para poder editar o leer en la base de datos de EvilCorp y en ninguna otra. Al final de todo, haremos un flush de los privilegios para que se actualicen los permisos de estos usuarios recién creados.

COMPROBACIONES

Juntando todo esto, se nos queda el script de esta forma:

```
1    create database if not exists EvilCorp;
2    drop table if exists `users`;
3    create table `users`(id_user BIGINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY, username VARCHAR(32) NOT NULL, name VARCHAR(64) NOT NULL, surname VARC HAR(64), birthday DATE NOT NULL, email VARCHAR(32) NOT NULL, phone VARCHAR(1 5) NOT NULL, country VARCHAR(64) NOT NULL, direction VARCHAR(64), password V ARCHAR(32), date DATETIME NOT NULL DEFAULT now());
4    insert into `users`(username, name, surname, birthday, email, phone, country , direction, password) values ("root", "Hilon", "Musgo", "1971-06-28", "hilo nmusgo@gmail.com","27666666666","South Africa","EvilCorp666","e1e71757deb074 60abff6678e3cd468f");
```

```
5    create user 'editor'@'localhost' identified by 'enti';
6    create user 'reader'@'localhost' identified by 'enti';
7    grant insert, update on EvilCorp.* to 'editor'@'localhost';
8    grant select on EvilCorp.* to 'reader'@'localhost';
9    flush privileges;
```

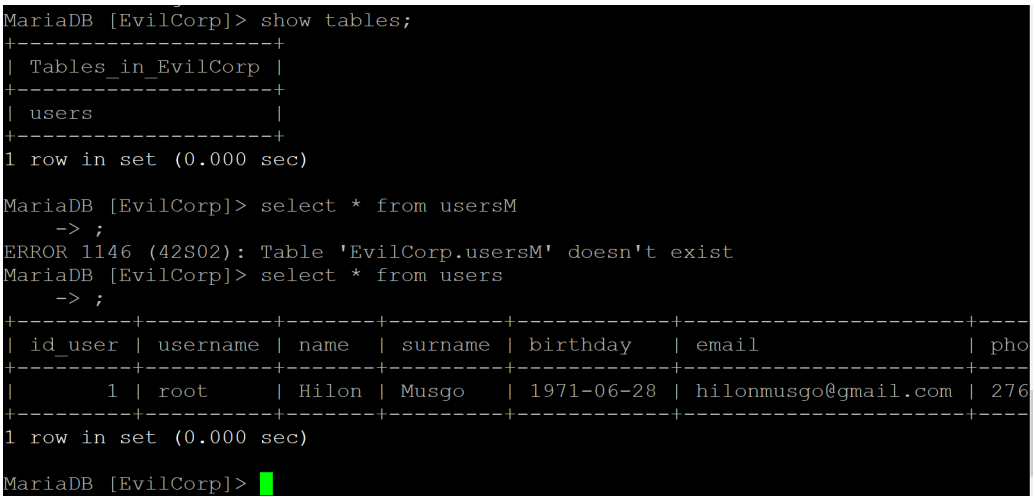
Ahora, es hora de ejecutar el script y ver los resultados. Para comprobarlo, lo que haremos será leer el script y pasarlo a mysql para que lo lea y ejecute las líneas del script. Luego de hacer eso, lo que haremos será entrar en Mysql y mostrar todas las bases de datos. Cuando lo ejecutamos y vamos a mysql para ver si esta el servidor, podemos ver que si que esta.

```
root@Ian:/home/enti/EvilCorp# cat database.sql | mysql EvilCorp
root@Ian:/home/enti/EvilCorp#
```

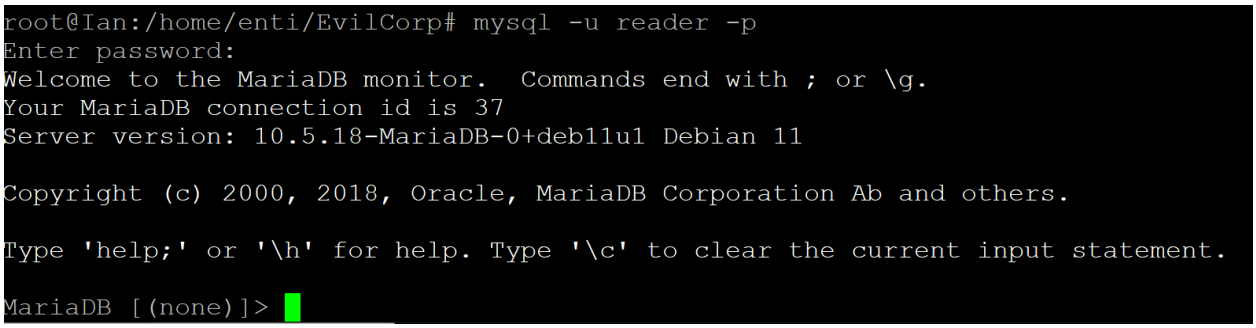
```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| BuscaPaco          |
| EvilCorp           |
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
5 rows in set (0.000 sec)
```

Luego, dentro de la base de datos, haremos dos comprobaciones, si se ha creado la tabla de users, y si se ha insertado el usuario de Hilon Musgo. A la hora de comprobar si esto ha funcionado, lo haremos seleccionando la base de datos de

EvilCorp y luego mostrando las tablas que hay, si existe la table de users mostraremos todo lo que hay en la tabla. Luego de hacer esa comprobación, podemos ver que en efecto, todo ha ido correctamente.



Aquí podemos ver como en efecto, está la tabla de users, y como en efecto, se ha insertado el usuario correctamente. Ahora comprobaremos si los usuarios del propio mysql se han creado correctamente. Para hacer esta comprobación lo que haremos será entrar como los dos usuarios y primero con el usuario reader leer la tabla, y luego con el usuario editor hacer un update de Hilon Musgo.



De primeras, lo que podemos comprobar es que nos deja entrar como el usuario, por lo que ahora probaremos si podemos hacer un select de la tabla users.

```
MariaDB [(none)]> use EvilCorp;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [EvilCorp]> select * from users;
+---------+----------+-------+---------+------------+----------------------+----
---------+--------------+-------------+----------------------------------+------
---------------+
| id_user | username | name  | surname | birthday   | email                | pho
ne       | country      | direction   | password                         | date
              |
+---------+----------+-------+---------+------------+----------------------+----
---------+--------------+-------------+----------------------------------+------
---------------+
|       1 | root     | Hilon | Musgo   | 1971-06-28 | hilonmusgo@gmail.com | 276
66666666 | South Africa | EvilCorp666 | e1e71757deb07460abff6678e3cd468f | 2023-
05-01 21:31:07 |
+---------+----------+-------+---------+------------+----------------------+----
---------+--------------+-------------+----------------------------------+------
---------------+
1 row in set (0.000 sec)

MariaDB [EvilCorp]>
```

Como podemos ver, nos deja hacer un select, ahora comprobaremos si podemos hacer un update con el usuario editor.

```
root@Ian:/home/enti/EvilCorp# mysql -u editor -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 39
Server version: 10.5.18-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Como podemos ver, podemos acceder al usuario de editor. Ahora miraremos si podemos hacer un update con ese usuario.

```
MariaDB [EvilCorp]> update users set name = "Jilon";
Query OK, 1 row affected (0.002 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [EvilCorp]> select * from users;
ERROR 1142 (42000): SELECT command denied to user 'editor'@'localhost' for table
 `EvilCorp`.`users`
MariaDB [EvilCorp]>
```

Aquí podemos ver como nos deja hacer update, pero no select, por lo que los permisos funcionan.
Con estas comprobaciones podemos decir que el script para la base de datos funciona correctamente.

4.HARDWARE

REQUERIMIENTOS

Primero, para saber que tipo de hardware necesitaremos, tendremos que tener en cuenta las necesidades de nuestro servidor. En este caso, para sostener la base de datos de usuarios de Evil Corp tendremos que tener en cuenta cuánto ocuparan los datos de un solo usuario en nuestra base de datos.
Para realizar esta tarea tendremos que saber las tablas de nuestra base de datos y los campos de dichas tablas. Como en este caso solo tenemos una tabla de usuarios, solo nos hará falta comprobar los campos de dicha tabla. La tabla en concreto contiene, el conteo de bytes que ocupa cada campo es suponiendo que se rellenen a su capacidad máxima:

ID= *El identificador de los usuarios ocupa una cantidad de 900 bytes ya que mariadb guarda dicha llave primaria con este espacio de base.*

Username= *El campo del nombre de usuario ocupa una cantidad de 128 bytes ya que el máximo de caracteres de dicho campo son 32 pero MariaDB usa la codificación de UTF8mb4, y esta codificación*

*aumenta el tamaño a 4 por carácter. Esto hace que sean 32*4 bytes.*

Name= *El campo del nombre del cliente ocupa una cantidad de 256 bytes ya que el máximo que puede almacenar son 64 caracteres y con la codificación UTF8mb4 aumenta 4 veces su tamaño.*

Surname= *El campo de apellido del cliente ocupa una cantidad de 256 bytes ya que el máximo que puede almacenar son 64 caracteres y con la codificación UTF8mb4 aumenta 4 veces su tamaño.*

Birthday= *El campo de nacimiento ocuparía un total de 3 bytes ya que es un byte por campo, el campo de día, mes y año.*

Email= *El campo de correo electrónico del cliente ocuparía un total de 128 bytes ya que tiene como máximo 32 caracteres y por la codificación UTF8mb4 un carácter ocupa 4 veces más.*

Phone= *El número de teléfono del cliente ocuparía 15 bytes ya que el numero de telefono mas largo del mundo ocupa 15 caracteres.*

Country= *El país del cliente ocuparía 256 bytes ya que el campo tendrá como máximo 64 caracteres y con la codificación UTF8mb4 dichos caracteres ocuparan 4 veces más.*

Direction= *La dirección del cliente ocuparía 256 bytes ya que el campo tendrá como máximo 64 caracteres y con la codificación UTF8mb4 dichos caracteres ocuparan 4 veces más.*

Password= *La contraseña ocuparía en todos los casos 128 bytes ya que es un campo de 32 caracteres los cuales salen de una*

encriptación con MD5, por lo que todas las contraseñas serán de 32 caracteres.

Contando todos estos campos junto con lo que ocupan en bytes tendríamos, suponiendo que se rellenan todos los campos a su máxima capacidad, una cantidad de 1382 TB si toda la población mundial rellena sus datos en esta base de datos. Como necesitamos asegurarnos de que esta base de datos siga funcionando y que no se colapse tomaremos medidas y comprobaremos cuánto ocuparía si 2 mil millones de personas más rellena este formulario. El resultado que nos da son 2368 TB de memoria que necesitariamos.
Para abaratar costes supondremos que no toda la población tendrá nombres de 32 caracteres, ni apellidos de 64, ni que viven en un país de 64 caracteres ni tendrán un correo electrónico de 32 caracteres ni que su dirección tenga 64 caracteres. Por eso reduciremos a la mitad estos campos para el cálculo del espacio. Como resultado nos sale que necesitaremos 1728 TB de espacio. Con este número procederemos a pensar en el hardware que necesitaremos.
También tendremos que tener en cuenta que utilizaremos el sistema de RAID 6 para el almacenamiento de la base de datos, por lo que hay que contar con discos duros y ssd extra.

COMPONENTES DEL HARDWARE

A la hora de elegir los componentes tendremos que tener en cuenta el precio, la cantidad que necesitaremos, su compatibilidad y su rendimiento.

PROCESADOR: *La elección del procesador será el AMD EPYC™ 7763. Elegiremos este procesador, el cual es de tercera generación, porque para realizar una tarea del calibre de esta base de datos es necesario mucho procesado. También es importante que para esta tarea es más importante la cantidad de núcleos y de hilos antes que la potencia como tal, es por eso que este procesador con 64 núcleos y 128 hilos es el que elegiremos.*

RAM: *La elección para las tarjetas ram será el Samsung DDR4-3200 ECC LRDIMM. Elegiremos esta ram ya que tiene mucha capacidad, un total de 64GB de ram, y tiene una velocidad de 3200 MHz. Para la tarea de este servidor será necesaria mucha memoria de rápido acceso ya que se manejan muchos datos en un periodo de tiempo muy corto. También es una tarjeta ram la cual es compatible con el procesador. Es por esto que esta tarjeta ram será la que elegiremos.*

Placa Base: *La elección de la placa base es la Supermicro H12DSG-NT. Es esto ya que es una placa de un tamaño aceptable, es compatible con la tarjeta ram, y es capaz de ejercer la función de servidor dual con dos procesadores de tercera generación de la gama AMD EPYC, es por eso que usaremos esta placa.*

Disco Duro(HDD): *Para el disco duro elegiremos el Seagate Exos X16 3.5" 16TB SATA3. Elegiremos este disco duro ya que tiene una capacidad de 16TB y tiene una velocidad de rotación de 7200 RPM. Es por esto que serán los discos duros para esta base de datos y para las copias de seguridad.*

Unidad de Estado Sólido(SSD): *La elección para la ssd de nuestro servidor será el Samsung PM1733. Elegiremos este ssd ya que tiene una capacidad de 4TB y al ser una unidad rápida de por sí, es muy útil para la tarea de esta base de datos.*

CONFIGURACIÓN DISCOS DUROS

La configuración que tendremos en los discos duros será la de RAID 6. Esta configuración lo que hace es distribuir los datos en un conjunto de discos físicos. Estos discos físicos son 4 normalmente y trabajan juntos para proporcionar mayor velocidad y fiabilidad ya que los datos se distribuyen entre los discos en forma de segmentos. En el sistema RAID 6 se usan dos unidades de paridad las cuales no almacenan los datos reales, sino que se usan para calcular los datos que falten en el momento de recuperar algo. Este sistema es más costoso pero más seguro, y a la hora de gestionar una base de datos tan extensa la mejor opción es el RAID 6. A la hora de aplicar este sistema en nuestro servidor lo que haremos será poner en cada máquina los 4 discos para el sistema RAID 6 junto a los 12 discos duros restantes que servirán para almacenar los datos ya que en una placa Supemicro H12DSG-NT tiene 16 puerto sata3.

PRESUPUESTO DE LA BASE DE DATOS

En el presupuesto se toman en cuenta los componentes de la base de datos, el cluster y las copias de seguridad.

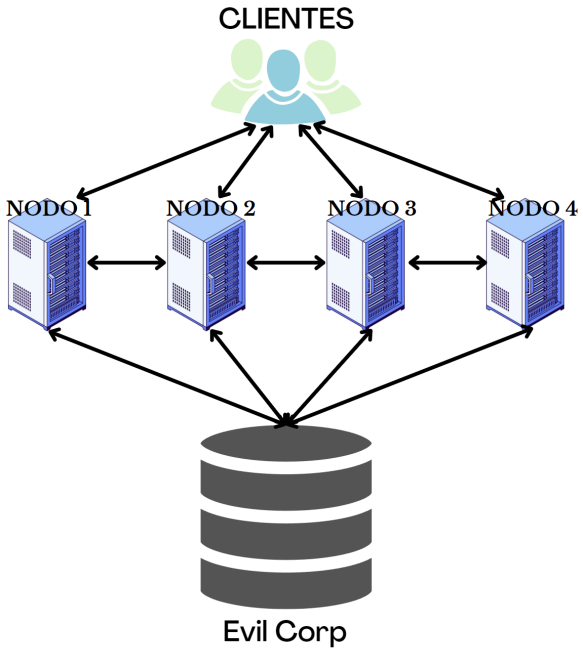
COMPONENTE	*CANTIDAD*	*PRECIO INDIVIDUAL*	*PRECIO TOTAL*
AMD EPYC™ 7763	50	7164€	358.200€
Samsung DDR4-3200 ECC LRDIMM	1920	151€	289.920€
Supermicro H12DSG-NT	30	1071€	32.130€
Seagate Exos X16 3.5" 16TB SATA3	480	300€	144.000€
Samsung PM1733	24	389€	9.336€
RACKS	30	500€/1000€	MIN: 15000€ MAX: 30000€
PRECIO TOTAL	N/A	N/A	MIN: 848.586€ MAX: 863.586€

5.CLUSTER

DISEÑO CLUSTER

El cluster que usaremos será un cluster de balanceo. Un cluster de balanceo es un conjunto de servidores que trabajan juntos para poder distribuir entre sí la carga de trabajo de manera equilibrada. En este tipo de cluster las máquinas que se encargan de la distribución de la carga entre el propio cluster y los clientes se llaman distribuidores de carga. Gracias a este tipo de cluster podremos tener el servicio en constante movimiento y si falla una máquina momentáneamente podremos pasarle la carga de trabajo a las demás máquinas del cluster.

En nuestro caso lo que tendremos serán 4 nodos de distribución de carga que estarán conectados entre ellos por si uno se ve sobrecargado pasar la carga a otro. Estos nodos de distribución estarán conectados al cliente y al servidor de Evil Corp para controlar la carga entre los servidores, donde se encuentra la base de datos, y el cliente, quien realiza la petición al servidor. Un diseño muy visual de cómo sería:



Este cluster nos permitirá tener todas las máquinas activas con una carga equilibrada para que no ocurra ninguna sobrecarga en ningún módulo. Gracias a esto podremos evitar que los clientes del servidor sufran expulsiones repentinas o cuelgues del servicio y podremos evitar que las máquinas se rompan por el sobreesfuerzo.

6.COPIAS DE SEGURIDAD

¿Qué hardware compraréis para ayudaros?

Para ayudar a la hora de hacer copias de seguridad tenemos varias opciones.

La primera de todas, las unidades de cintas. Estas cintas son una opción viable ya que son muy duraderas y tienen mucha capacidad de almacenaje. La única pega es que son mucho mas lentas que los discos duros normales.

La segunda opción es utilizar sistemas de almacenamiento de red como NAS o SAN ya que son locales y de alta velocidad. Sirven para interconectar los servidores y poder tener una "nube" de forma local.

La tercera opción sería tener un sistema de discos duros externos donde guardar estas copias de seguridad. Estos discos duros externos son fáciles de usar y fáciles de conectar a los servidores por lo que es una opción viable.

¿Cómo y cuándo se realizan las copias de seguridad?

Las copias de seguridad se harán diarias, semanales y mensuales. Estas copias de seguridad se almacenarán de forma externa. Las copias diarias, semanales y mensuales se eliminarán una vez que se haga una nueva copia, por lo que siempre habrá espacio y una copia a la que acudir si hay algún fallo o alguna pérdida importante. Los datos críticos se guardarán en cintas.

¿Qué ocurre si falla un disco duro y cómo debemos recuperarlo? ¿Cómo entra en juego la configuración de los discos duros?

Cuando un disco duro falla lo que ocurrirá será que gracias al sistema RAID 6 podremos salvar el contenido de dicho disco duro y cambiarlo con facilidad y gracias al cluster de balanceo podremos derivar el tráfico a otro servidor para que siga todo en funcionamiento.

¿Y si falla una máquina entera?

Cuando falle una máquina lo que harán los distribuidores de carga será intentar equilibrar las peticiones entre las máquinas restantes para que no se cuelgue el servicio mientras se reemplaza la siguiente máquina. Hay un total de 10 máquinas de reemplazo por si ocurre algún fallo. Dichas máquinas de repuesto tienen el mismo software que las originales para evitar problemas de compatibilidad a la hora de intercambiarla.

7.CRON

SCRIPT DE LAS COPIAS DE SEGURIDAD

Primero de todo, lo que más necesitaremos será el script que se encargue de las copias de seguridad. Dicho script tendrá que tener las tres copias de seguridad, la diaria, la semanal y la mensual. A su vez, el script tendrá que comprobar si hay más de un archivo en la carpeta de las copias de seguridad y si hay más de uno, eliminar el antiguo. Luego tendrá que comprimir el archivo en un tar.gz con la fecha de cuando se realizó la copia de seguridad. El script en concreto será este:

```
1   !/bin/bash
2   DAY=`echo $(date +%F)`
3   WEEK=`echo $(date +%F_%V)`
4   MONTH=`echo $(date +%F_%m)`
5   FILE_LIST_WEEK=`ls /root/backups/weekly/`
6   FILE_LIST_MONTH=`ls /root/backups/monthly/`
7   ROUTE_DAILY="/root/backups/daily/"
8   ROUTE_WEEK="/root/backups/weekly/"
9   ROUTE_MONTH="/root/backups/monthly/"
10
11  rm $ROUTE_DAILY*
12  mysqldump EvilCorp > ${ROUTE_DAILY}backup.sql
13  tar  -czvf     ${ROUTE_DAILY}backup_$DAY.tar.gz
${ROUTE_DAILY}backup.sql
14  rm ${ROUTE_DAILY}backup.sql
15
16  for FILE_NAME_WEEK in $FILE_LIST_WEEK
17  do
```

```
 18  if [ $FILE_NAME_WEEK != "backup_${WEEK}" ]
 19  then
 20  rm $ROUTE_WEEK*
 21  mysqldump EvilCorp > ${ROUTE_WEEK}backup.sql
22   tar  -czvf     ${ROUTE_WEEK}backup_$WEEK.tar.gz
${ROUTE_WEEK}backup.sql
 23  rm ${ROUTE_WEEK}backup.sql
 24  fi
 25  done
 26
 27
 28  for  FILE_NAME_MONTH in $FILE_LIST_MONTH
 29  do
 30  if [ $FILE_NAME_MONTH != "backup_${MONTH}" ]
 31  then
 32  rm $ROUTE_MONTH*
 33  mysqldump EvilCorp > ${ROUTE_MONTH}backup.sql
 34  tar  -czvf     ${ROUTE_MONTH}backup_$MONTH.tar.gz
${ROUTE_MONTH}backup.sql
 35  rm ${ROUTE_MONTH}backup.sql
 36  fi
 37  done
```

El script tiene las variables ***day, week*** *y* ***month****. Estas variables guardaran la fecha en formato dd-mm-yyyy añadiendo al final una barra baja con el mes en el que se hace la copia de seguridad o con la semana. Esto sirve para poder identificar con más facilidad las copias de seguridad y así poder facilitar el script. Las variables* ***FILES_LIST_WEEK*** *y* ***FILES_LIST_MONTH*** *son para guardar la lista de archivos de las carpetas week y month, que son los directorios donde se guardan las copias de seguridad. Las variables* ***ROUTE_DAILY, ROUTE_WEEKLY*** *y*

ROUTE_MONTHLY *guardaran las rutas de los directorios donde se almacenarán las copias de seguridad.*

Luego, el script comienza eliminando, haciendo una copia de seguridad, comprimiendo la copia de seguridad diaria y luego eliminando la copia que no está comprimida. Esto se hace así ya que como se hace de manera recurrente cada dia, no hará falta hacer comprobaciones, simplemente eliminar todo lo de dentro y hacer una copia nueva. Primero se hace el ***rm*** *de todo lo que haya en la ruta especificada, luego se hace el* ***mysqldump*** *de la base de datos para pasarla en su carpeta correspondiente, luego se comprime en un tar.gz con el comando* ***tar -czvf*** *y luego elimina la copia no comprimida. Los parámetros del comando* ***tar*** *lo que hacen es:*

c: *Crea un archivo tar nuevo.*

z: *Utiliza el comando gzip para comprimir el archivo.*

v: *Muestra más información por pantalla.*

f: *Enseña el nombre del archivo.*

Las siguientes líneas muestran un bucle ***for*** *donde busca en los archivos del directorio* ***weekly*** *para luego hacer un* ***if*** *donde comprobará si el archivo que hay dentro es de la misma semana o es de otra. Si es de otra semana borrara el archivo actual y entonces hará el mismo proceso que con la copia de seguridad diaria. El siguiente* ***for*** *realiza la misma acción que el anterior pero con la carpeta de* ***monthly****.*

Si fuera un script realista para la base de datos de EvilCorp tendría que mandar las copias de seguridad a los discos duros externos, a la "nube" que tendríamos con la red ***NAS*** *o a las cintas. Estas copias de seguridad se almacenaron en varios discos para evitar posibles fallos o ransomwares.*

LINEAS DEL CRONTAB

```
  1  # /etc/crontab: system-wide crontab
  2  # Unlike any other crontab you don't have to run the
`crontab'
  3  # command to install the new version when you edit this
file
  4  # and files in /etc/cron.d. These files also have username
fields,
  5  # that none of the other crontabs do.
  6
  7  SHELL=/bin/sh
  8
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bi
n
  9
 10  # Example of job definition:
 11  # .---------------- minute (0 - 59)
 12  # |  .------------- hour (0 - 23)
 13  # |  |  .---------- day of month (1 - 31)
 14  # |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
 15  # |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR
sun,mon,tue,wed,thu,fri,sat
 16  # |  |  |  |  |
 17  # *  *  *  *  * user-name command to be executed
 18
 17  *       *  *  *     root       cd  /  &&  run-parts  --report
/etc/cron.hourly
 19  25 6     * * *   root    test -x /usr/sbin/anacron || ( cd /
&& run-parts --report /etc/cron.daily )
 20  47 6     * * 7   root    test -x /usr/sbin/anacron || ( cd /
&& run-parts --report /etc/cron.weekly )
```

```
 21  52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd /
&& run-parts --report /etc/cron.monthly )
 22  0  2    * * *   root    /home/enti/buscaBackup.sh
 23  0  5    * * *   root    /root/backups/.backup_script.sh
 24  #
```

La línea que nos interesa es la penúltima. Esa línea nos indica que cada día a las 5 de la mañana se ejecutará el script de nuestras copias de seguridad.

8.COMPRESION

Para comprimir nuestros scripts primero debemos usar el comando tar con los parámetros -czvf, luego declarar el archivo y añadir el .tar.gz y finalmente señalizar la carpeta donde irá el archivo comprimido.

```
tar -czvf backup.sh.tar.gz /home/enti/EvilCorp/
tar -czvf database.sql.tar.gz /home/enti/EvilCorp/
```