Configurar UDS Enterprise 3.5 en alta disponibilidad

www.udsenterprise.com

Introducción

UDS Enterprise permite realizar la configuración de sus diferentes componentes en alta disponibilidad (HA). Este modo de configuración permite dotar al entorno VDI de continuidad ante el fallo de algún nodo de virtualización o por al fallo del propio S.O. de alguno de los componentes del entorno.
.

Para dotar al entorno VDI de una alta disponibilidad completa, además de configurar varias máquinas UDS-Server y UDS-Tunnel, también será necesario disponer de una replicación o configuración en cluster de la base de datos a la que se conectan los servidores UDS. Otro elemento necesario y que también tendremos que configurar en alta disponibilidad, será el balanceador de carga que gestione y reparta las diferentes conexiones a los componentes UDS-Server y UDS-Tunnel.

UDS Enterprise soporta balanceadores de tipo físico (ej: F5) o de tipo virtual (ej: HAProxy), estos tienen que tener soporte para modos TCP y HTTP.

En el presente documento, a través de un ejemplo completo de configuración, trataremos de abordar todos los pasos para configurar UDS Enterprise en Alta Disponibilidad, desde los elementos propios de UDS (UDS-Server y UDS-Tunnel) hasta un balanceador de carga software (HAProxy) y una Base de datos MySQL.

Elementos necesarios

En esta guía utilizaremos los componentes necesarios para la mayoría de los despliegues de un entorno UDS en HA. Son los siguientes:

Servidores de base de datos MySQL

Los servidores de base de datos (BBDD) que utilizaremos serán los proporcionados por el equipo de UDS. En estos servidores se guardarán todos los registros y configuraciones de UDS.

En este documento mostramos la configuración de dos servidores MySQL, uno principal y otro secundario (Master y Slave), en modo de replicación activo/pasivo.

NOTA:
A partir de la versión 3.0 de UDS Enterprise, se soportan configuraciones de clusters MySQL activo/activo.
El componente de base de datos es unos de los componentes más importantes del entorno VDI con UDS. Por tanto, para despliegues en producción se recomienda encarecidamente disponer de respaldo en este componente, ya sea vía backup de máquina completa, instancia de BD utilizada en UDS, configuración en cluster, o como se mostrará en este documento, una configuración de réplica activo/pasivo.

Servidores HAProxy

Será el servidor encargado de balancear las conexiones de los servidores UDS Server y Tunnel. A través de él se realizará el acceso de usuarios/administradores en el portal de login de UDS y las conexiones a los diferentes servicios.

En este documento se configuran dos máquinas HAProxy, en modo activo/pasivo.

NOTA:
En los diferentes servidores HAProxy configuraremos una dirección IP que estará activa solamente en el servidor principal.En caso de caída o aislamiento de este servidor, se activará automáticamente en los otros servidores secundarios HAProxy.

Servidores UDS-Server

Podremos añadir todas las máquinas UDS-Server que necesitemos y hacerlas funcionar en modo activo/activo. Esto permitirá acceso continuo al portal de login a usuarios y administradores aunque perdamos alguna de las máquinas UDS-Server.

En este documento se configuran dos máquinas UDS-Server, en modo activo/activo.

Servidores UDS-Tunneler

Podremos añadir todas las máquinas UDS-Tunnel que necesitemos y hacerlas funcionar en modo activo/activo, esto permitirá acceso a servicios (escritorios o aplicaciones) a través de conexiones tunelizadas y HTML5 aunque perdamos alguna de las máquinas UDS-Tunnel.

En este documento se configuran dos máquinas UDS-Tunnel, en modo activo/activo.

NOTA:
Si un usuario está conectado a un servicio (escritorio o aplicación) y cae el servidor tunnel por el que está conectado, la conexión se perderá. Pero al volver a realizar la conexión, recuperará acceso al servicio a través de otro servidor tunnel activo de forma automática.

Requisitos para el despliegue

En este ejemplo de configuración de UDS Enterprise en HA, se han utilizado los siguientes recursos:

MySQL:

- 2 servidores MySQL (proporcionados por el equipo de UDS Enterprise). Los requisitos mínimos para cada máquina son: 2 vCPUs, 1 GB de vRAM y 8 GB de disco
- Datos IP: 2 direcciones IP, una para cada servidor (Master - Slave), máscara de red, Gateway y DNS.
- Datos BBDD: Instancia, usuario y contraseña (por defecto, instancia: uds, usuario: uds, contraseña: uds).

HAProxy:

- 2 máquinas con S.O. Linux Debian (puede utilizar servidores preconfigurados proporcionados por UDS disponibles en este repositorio: http://images.udsenterprise.com/files/UDS_HA/HAProxy/3.5/OVA-3.5/) con al menos 2 vCPUs, 1 GB de vRAM, 10 GB de disco.
- Datos IP: 3 direcciones IP, una para cada servidor (Master - Slave) y una IP virtual compartida entre los dos servidores que servirá para el balanceo), máscara de red, gateway y DNS.
- Acceso a internet.
- Certificado: Es necesario disponer (o generar) un certificado válido para las conexiones SSL en formato PEM. En este ejemplo se muestra cómo crear un certificado temporal.

UDS-Server:

- 2 máquinas UDS-Server (proporcionados por el equipo de UDS Enterprise). Los requisitos mínimos por cada máquina son: 2 vCPUs, 2 GB de vRAM y 8 GB de disco.
- Datos IP: 2 direcciones IP, una para cada servidor, máscara de red, gateway y DNS.
- Número de serie válido.
- Datos de conexión con la BBDD MySQL: dirección IP, instancia, usuario y contraseña.

UDS-Tunnel:

- 2 máquinas UDS-Tunnel (proporcionados por el equipo de UDS Enterprise). Los requisitos mínimos por cada máquina son: 2 vCPUs, 2 GB de vRAM y 10 GB de disco.
- Datos IP: 2 direcciones IP, una para cada servidor, máscara de red, gateway y DNS.
- Dirección IP de balanceo de los servidores HAProxy.

Configuración de los servidores MySQL

Nos validaremos en los servidores de base de datos facilitados por el equipo de UDS Enterprise utilizando las credenciales:

- **Usuario:** root
- **Contraseña:** uds

Configuraremos el nuevo nombre DNS de los servidores con el comando:

`hostnamectl set-hostname` ***nombre_servidor***

Y realizaremos la configuración IP de las máquinas MySQL, a través del fichero:

/etc/network/interfaces

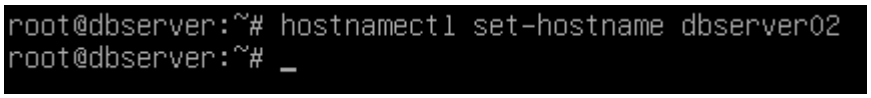
Nodo principal (Master):

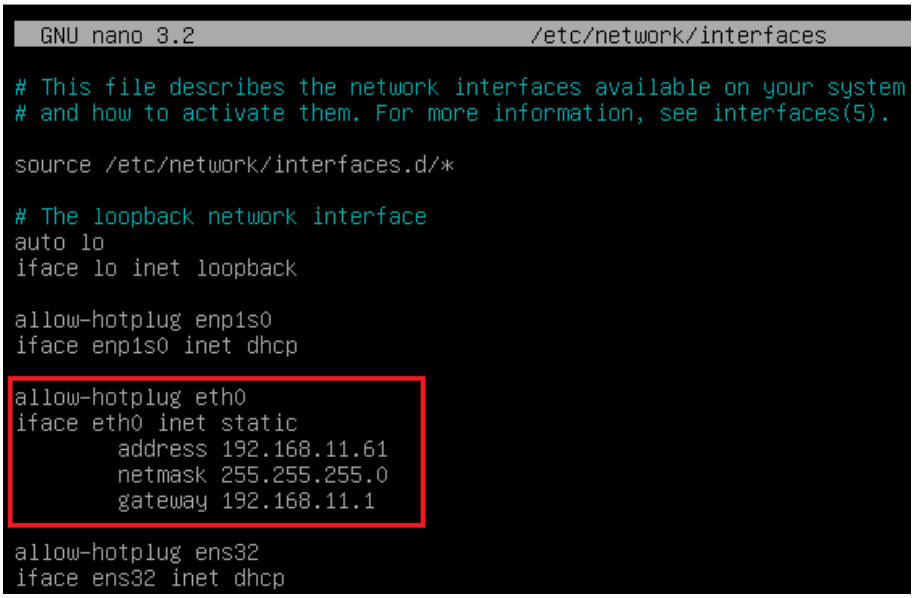
```
root@dbserver:~# hostnamectl set-hostname dbserver01
root@dbserver:~#
```

```
  GNU nano 3.2                          /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

allow-hotplug enp1s0
iface enp1s0 inet dhcp

allow-hotplug eth0
iface eth0 inet static
        address 192.168.11.60
        netmask 255.255.255.0
        gateway 192.168.11.1

allow-hotplug ens32
iface ens32 inet dhcp
```

Una vez realizadas estas configuraciones en el nodo principal de base de datos, reiniciaremos el servidor para aplicar los cambios.

Nodo secundario (Slave):



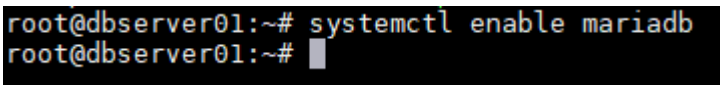


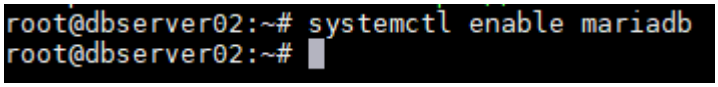
Una vez realizadas estas configuraciones en el nodo secundario de base de datos, reiniciaremos el servidor para aplicar los cambios.

La siguiente configuración no es obligatoria pero se recomienda realizarla en ambos servidores (Master - Slave)

Habilitamos el servicio MariaDB:

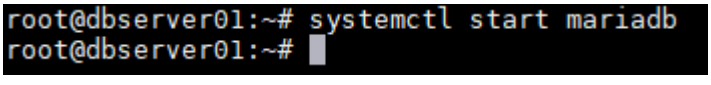
```
systemctl enable mariadb
```

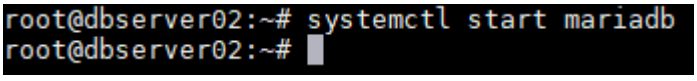




Iniciamos el servicio MariaDB:

```
systemctl start mariadb
```

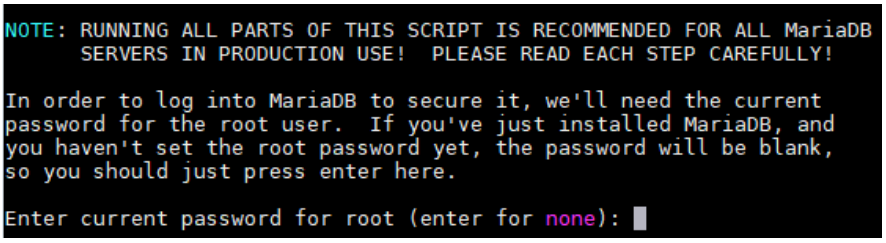




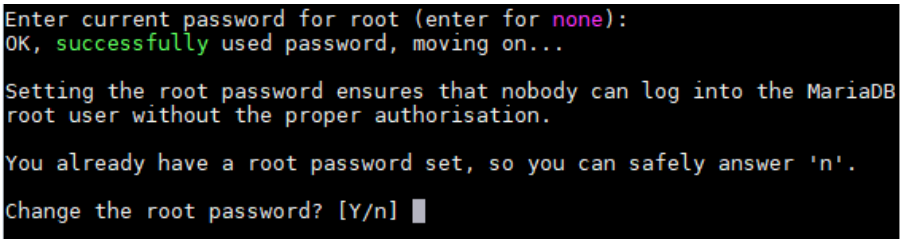
Lanzamos el script de configuración, para proteger nuestra base de datos:

```
mysql_secure_installation
```

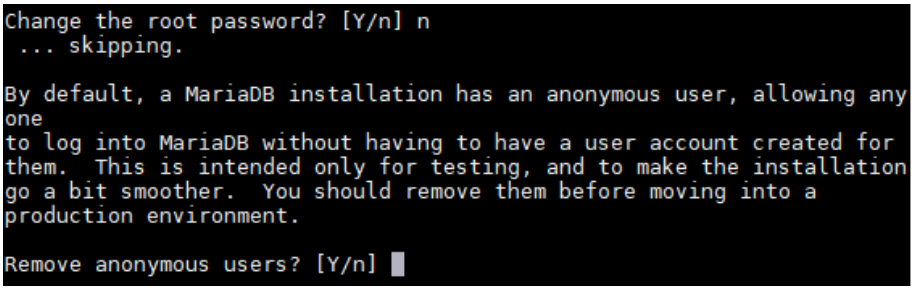
El asistente de instalación nos solicita que introduzcamos la contraseña actual para el usuario root, ya que para realizar el proceso necesitamos permisos de administrador.



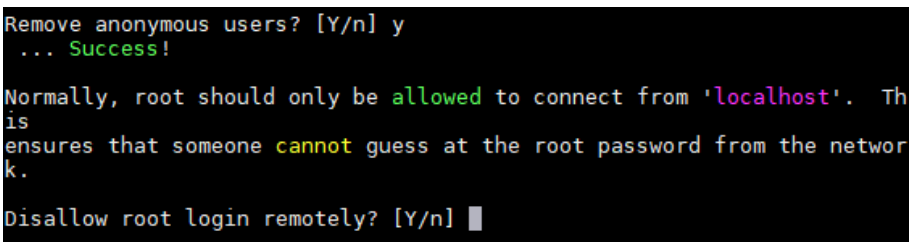
Nos preguntará si queremos cambiar la contraseña del usuario root. En este caso seleccionamos la opción: **NO**



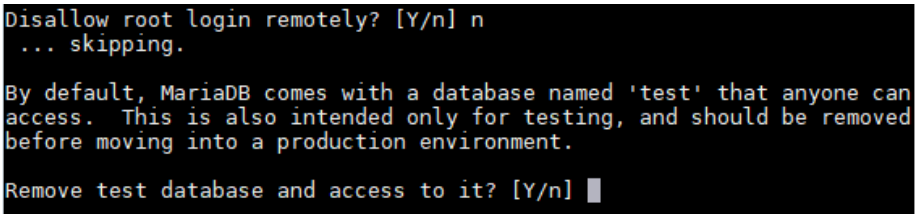
Nos preguntará si queremos eliminar los usuarios anónimos existentes. En este caso seleccionamos la opción: **Yes**



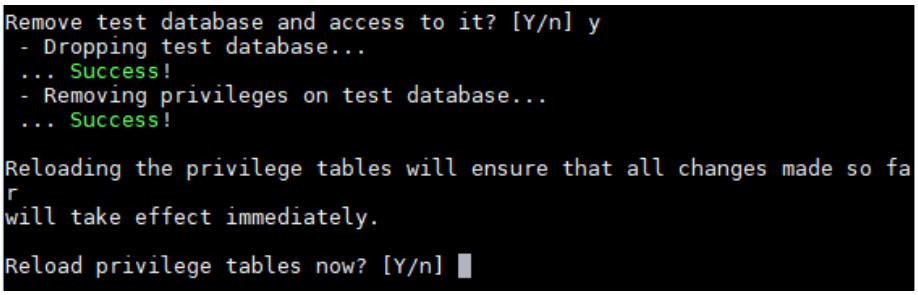
Nos preguntará si queremos deshabilitar el inicio de sesión del usuario root de forma remota.En este ejemplo seleccionamos la opción: **No**



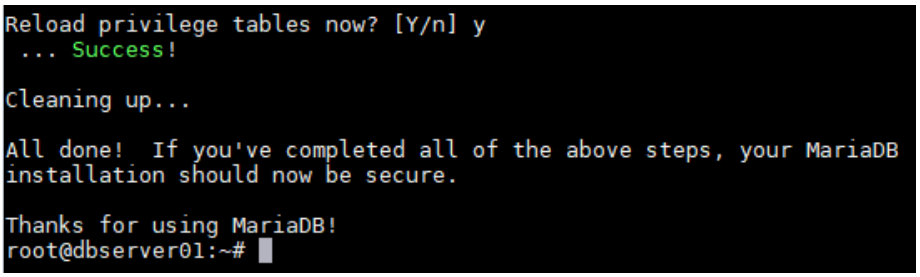
Nos preguntará si queremos eliminar la base de datos de prueba. En este ejemplo seleccionamos la opción: **Yes**



Nos preguntará si queremos recargar las tablas de privilegios. En este ejemplo seleccionaremos la opción: **Yes**



Tras completar el proceso en **ambos servidores**, procederemos a la siguiente tarea de configuración.



Configuración de la réplica entre servidores

Nodo principal (Master)

Editamos el fichero: /etc/mysql/mariadb.conf.d/50-server.cnf

En el parámetro: **bind-address** indicamos la dirección IP del servidor (en este caso la IP del servidor principal):

```
#skip-external-locking

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 192.168.11.60

#
# * Fine Tuning
```

Unas líneas más abajo, eliminamos el símbolo # y dejamos los parámetros: **server-id** y **log_bin** como se indica en la siguiente imagen:

```
# The following can be used as easy to replay backup logs or for replication.
# note: if you are setting up a replication slave, see README.Debian about
#       other settings you may need to change.
server-id               = 1
log_bin                 = /var/log/mysql/mysql-bin.log
expire_logs_days        = 10
#max_binlog_size         = 100M
#binlog_do_db            = include_database_name
#binlog_ignore_db        = exclude_database_name
```

Una vez modificado el fichero y salvados los cambios, reiniciamos el servicio de MySQL para aplicar los cambios:

```
root@dbserver01:~# systemctl restart mariadb
root@dbserver01:~#
```

Ahora crearemos un nuevo usuario para la replicación. Para ello accedemos a la consola MySQL con permisos de root:

```
root@dbserver01:~# mysql -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 275375
Server version: 10.5.12-MariaDB-0+deb11u1-log 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)]>
```

Una vez validados, ejecutaremos la siguiente sentencia para la creación del usuario:

```
CREATE USER 'replica'@'%' IDENTIFIED BY 'uds';
```

Donde "**replica**" será el nombre del nuevo usuario y "**uds**" su contraseña.

```
MariaDB [(none)]> CREATE USER 'replica'@'%' IDENTIFIED BY 'uds';
Query OK, 0 rows affected (0.001 sec)
```

A continuación, proporcionaremos el permiso "replication Slave" al usuario:

```
GRANT REPLICATION SLAVE ON *.* TO 'replica'@'%' IDENTIFIED BY 'uds';
```

```
MariaDB [(none)]> GRANT REPLICATION SLAVE ON *.* TO 'replica'@'%' IDENTIFIED BY 'uds';
Query OK, 0 rows affected (0.000 sec)
```

Por último, ejecutaremos el siguiente comando para obtener información del nombre del fichero binario y su posición:

```
SHOW MASTER STATUS\G
```

```
MariaDB [(none)]> SHOW MASTER STATUS\G
*************************** 1. row ***************************
            File: mysql-bin.000001
        Position: 666
    Binlog_Do_DB:
Binlog_Ignore_DB:
1 row in set (0.000 sec)
```

Tomaremos nota del nombre del fichero, en este caso: **mysql-bin.000001** y de su posición: **666**. Estos datos serán necesarios para la configuración del servidor secundario o Slave.

NOTA:
Los datos obtenidos pueden variar dependiendo de la instalación.

Nodo secundario (Slave)

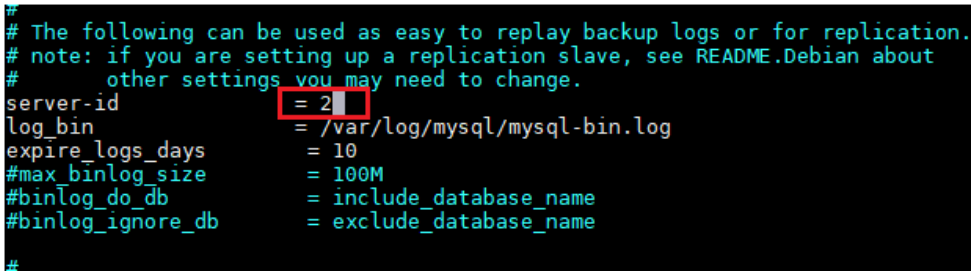
Procedemos a editar el mismo dichero de configuración que en el nodo principal, pero en lugar de indicar el valor **1** en el parámetro **server-id**, indicaremos **2.**

Editamos el fichero: /etc/mysql/mariadb.conf.d/50-server.cnf

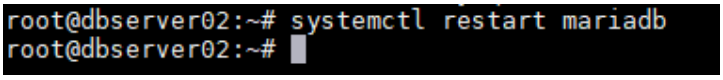
En el parámetro: **bind-address** indicamos la dirección IP del servidor (en este caso la IP del servidor secundario):

```
#skip-external-locking

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 192.168.11.61

#
# * Fine Tuning
```

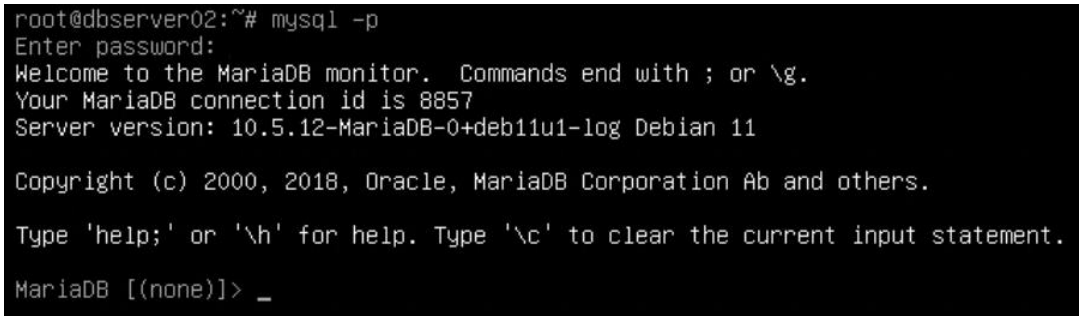
Unas líneas más abajo, eliminamos el símbolo # y dejamos los parámetros: **server-id** (en este caso, al ser el nodo secundario hay que cambiar el valor a **2**) y **log_bin,** como se indica en la siguiente imagen:



Una vez modificado el fichero y salvados los cambios, reiniciamos el servicio de MySQL para aplicar los cambios:

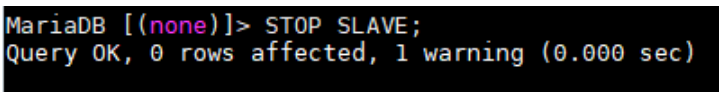


Ahora configuraremos los parámetros que utilizará el servidor secundario (Slave) para conectarse con el servidor principal (Master). Para ello accedemos a la consola MySQL con permisos de root:



Una vez validados, ejecutaremos el siguiente comando para parar cualquier operación en el servidor:

```
STOP SLAVE;
```



Una vez parado, ejecutaremos la siguiente sentencia para configurar la réplica entre el servidor principal y el servidor secundario:

```
CHANGE MASTER TO MASTER_HOST='192.168.11.60', MASTER_USER='replica',
MASTER_PASSWORD='uds', MASTER_LOG_FILE='mysql-bin.000001',
MASTER_LOG_POS=666;
```

Donde “**192.168.11.60**” será la dirección IP del servidor principal, “**replica**” el usuario de réplica configurado en pasos anteriores, “**uds**” la contraseña del usuario de réplica, “**mysql-bin.000001**” el nombre del fichero binario obtenido anteriormente del servidor principal y “**666**” la posición del fichero binario.

```
MariaDB [(none)]> CHANGE MASTER TO MASTER_HOST='192.168.11.60', MASTER_USER='replica', MASTER_PASSWO
RD='uds', MASTER_LOG_FILE='mysql-bin.000001', MASTER_LOG_POS=666;
Query OK, 0 rows affected (0.027 sec)
```

Iniciamos las operaciones en el servidor:

```
START SLAVE;
```

```
MariaDB [(none)]> START SLAVE;
Query OK, 0 rows affected (0.001 sec)
```

Para confirmar que la configuración realizada es correcta, ejecutamos el siguiente comando:

```
SHOW SLAVE STATUS\G
```

Confirmamos que la dirección IP del servidor principal es correcta y que “**Slave_IO_Running**” y “**Slave_SQL_Running**” están en “**Yes**”

```
MariaDB [(none)]> SHOW SLAVE STATUS\G;
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.11.60
                  Master_User: replica
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 666
               Relay_Log_File: mysqld-relay-bin.000002
                Relay_Log_Pos: 555
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
```

Probando la replicación

Podremos realizar una prueba sencilla para comprobar si la replicación configurada está activa y es correcta. Para ello, crearemos una nueva base de datos en el servidor principal y comprobaremos si de forma automática se replica en el servidor secundario:

1. Accedemos a la consola MySQL del servidor principal y creamos una nueva base de datos de test, llamada "**replicatest**":

```
CREATE DATABASE replicatest;
```

```
root@dbserver01:~# mysql -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 39
Server version: 10.3.22-MariaDB-0+deb10u1-log Debian 10

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)]> CREATE DATABASE replicatest;
Query OK, 1 row affected (0.000 sec)
```

2. Listamos las bases de datos para confirmar su correcta creación:

```
SHOW DATABASES;
```

```
MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| replicatest        |
| uds                |
+--------------------+
5 rows in set (0.001 sec)
```

3. Ahora accedemos a la consola MySQL del servidor secundario y confirmamos (con el comando: `SHOW DATABASES;`) que la base de datos creada anteriormente en el servidor principal ha sido replicada a este servidor (Slave):

```
MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| replicatest        |
| uds                |
+--------------------+
5 rows in set (0.011 sec)
```

Tras verificar que la replicación está funcionando, ya podremos conectar nuestros servidores UDS a la base de datos del servidor principal creada por defecto: “uds”

NOTA:
Para borrar la base de datos creada (“replicatest”) en el proceso de prueba, puede ejecutar, en el servidor principal, el siguiente comando:

```
DROP DATABASE replicatest;
```

Caída de los servidores

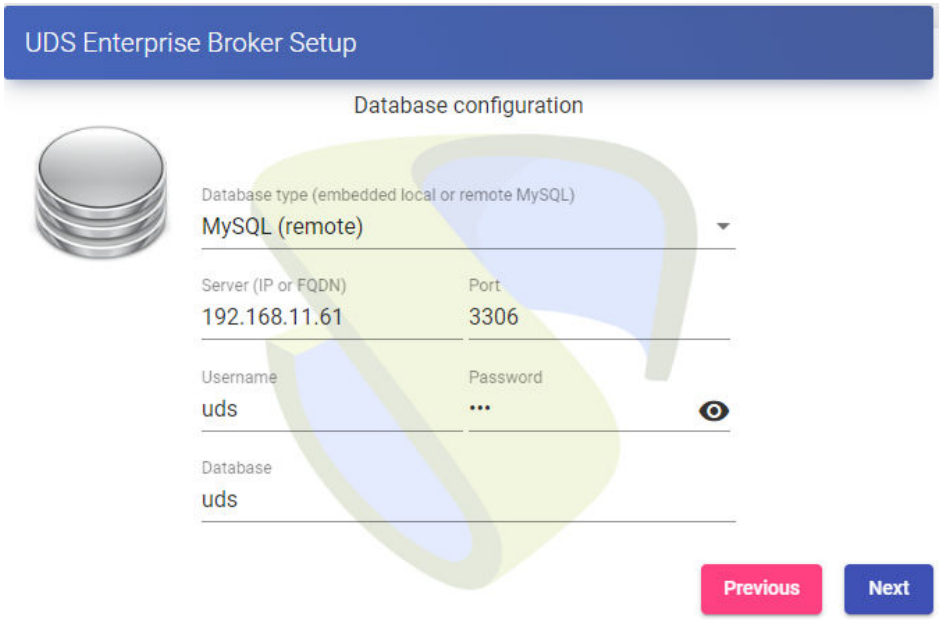
Si perdemos uno de los servidores de base de datos (ya sea por la caída del host de virtualización que lo aloja, por fallo del propio S.O., etc…), deberemos realizar una serie de tareas.

Dependiendo del servidor que perdamos (Master o Slave), las tareas que deberemos realizar, tanto para la continuidad del servicio VDI con UDS como para volver a tener una réplica activa, serán las siguientes:

Master (nodo principal)

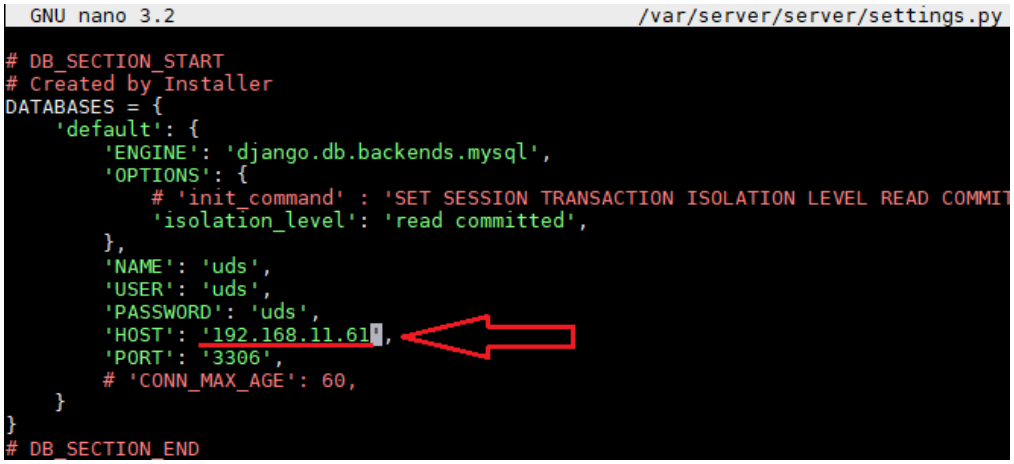
Si sufrimos la caída o pérdida del servidor de base de datos principal (Master), **perderemos el acceso** al entorno VDI y deberemos conectar de forma manual los diferentes servidores UDS-Server a la base de datos secundaria (Slave), la cual posee toda la información del entorno VDI hasta el momento de la caída del servidor principal.

Para hacer la nueva conexión con la base de datos en los servidores UDS, podremos ejecutar el asistente de configuración en las maquinas UDS-server (hay que realizarlo en todos los servidores UDS-Server). En el apartado de configuración de base de datos, indicaremos los nuevos datos (los datos del servidor Slave):



Otra opción, quizás más rápida y sencilla, para conectar con la nueva base de datos, sería editar el fichero de configuración en las máquinas UDS-Server (hay que realizarlo en todos los servidores UDS-Server) e indicar los datos de la nueva base de datos. El fichero de configuración está localizado en la siguiente ruta de la máquina UDS-Server:

/var/server/server/settings.py



Una vez modificada la dirección IP o nombre del nuevo host de base de datos, deberemos reiniciar el servidor. Esta tarea la repetiremos en todas las máquinas UDS-Server.

Reiniciado el servidor, ya volveremos a tener acceso al entorno VDI.

Ahora será necesario volver a dotar al sistema de otra máquina de réplica de base de datos. Para ello dispondremos de varias opciones, entre ellas:

- Configurar la actual máquina de base de datos como Master y generar una nueva máquina de réplica, la cual deberemos configurar y recuperar un backup con los datos existentes (puesto que solo se replicarán los datos nuevos).

- Directamente realizar un backup de la actual máquina de base de datos (parando previamente todas las máquinas UDS-Server). Habrá que generar una nueva máquina de base de datos Master, recuperando ahí el backup y volver a realizar la configuración de réplica.

NOTA:
Para no perder ningún dato, antes de aplicar cualquier método para reconstruir la replicación, se recomienda disponer de un backup de la base de datos para no perder ningún dato. Se puede utilizar el siguiente comando para realizar el backup:

```
mysqldump -u usuario -ppassword --databases instancia >
/ruta/nombre_dump.sql
```

Al realizar este backup es necesario que todas las máquinas UDS-Server se encuentren en estado apagado, de esta forma aseguramos la consistencia de datos y que no haya diferencia de datos entre el servidor Master y Slave antes de configurar la réplica.

Slave (nodo secundario)

Si sufrimos la caída o pérdida del servidor de base de datos secundario (Slave), **no perderemos el acceso** al entorno VDI pero deberemos volver a configurar un servidor de réplica Slave. Antes de realizar dicha configuración será necesario restaurar un backup con el actual estado de la base de datos principal, puesto que solo se sincronizaran los nuevos datos de réplica (no se replicarán los datos existentes en la base de datos).

Es importante que durante todo este proceso las máquinas UDS-Server estén apagadas para evitar que haya diferencias entre las BBDD de los servidores Master y Slave.

Configuración de los servidores HAProxy

En este documento se utilizarán los servidores HAProxy facilitados por el equipo de UDS Enterprise. Estos servidores están preconfigurados y solo será necesario modificar ciertos datos para tenerlos completamente configurados.

Los servidores los podremos descargar del siguiente repositorio:

https://images.udsenterprise.com/files/UDS_HA/HAProxy/3.5/OVA-3.5/

Ambos servidores están configurados con los siguientes recursos: 2 vCPUs, 1 GB de vRAM, 10 GB de disco y 1 vNIC.

Los servidores tienen un usuario creado: ***user***, con la contraseña: ***uds***. La contraseña del usuario root es: ***uds***

Una vez importados a la plataforma de virtualización, procederemos a su configuración

NOTA:
Estos servidores se facilitan en formato .OVA preparados para importar en entornos VMware. Si fuera necesario importarlos en otra plataforma de virtualización diferente, se puede extraer (ej: Winrar) su disco .vmdk y convertir (ej: qemu.img) al formato de la plataforma destino.

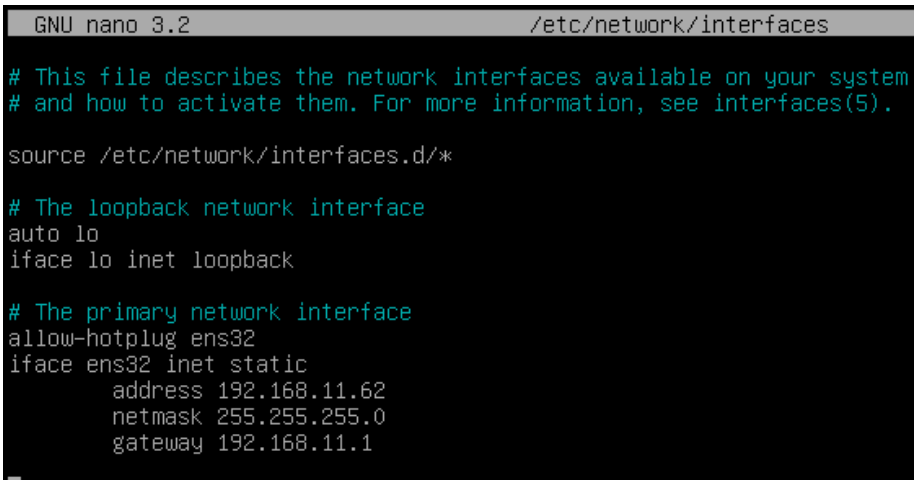
Se recomienda encarecidamente modificar la contraseña por defecto por una de mayor seguridad.

- **TAREAS A REALIZAR EN EL SERVIDOR HAPROXY PRINCIPAL**

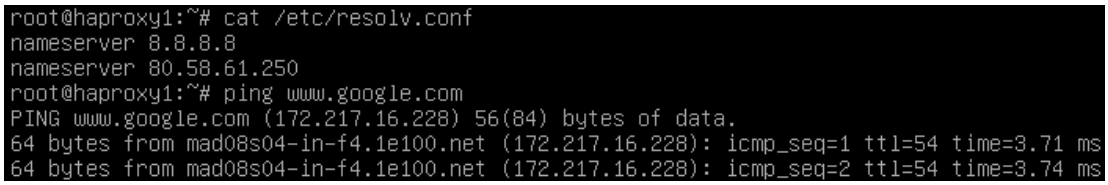
Una vez importada la máquina a la plataforma virtual y encendida, deberemos validarnos con el usuario: ***root*** y la contraseña: ***uds***

```
Debian GNU/Linux 11 haproxy1 tty1

haproxy1 login: root
Password:
Linux haproxy1 5.10.0-9-amd64 #1 SMP Debian 5.10.70-1 (2021-09-30) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Nov 22 12:58:19 CET 2021 from 192.168.11.2 on pts/0
root@haproxy1:~#
```

Configuraremos los nuevos datos IP modificando el fichero: /etc/network/interfaces



Y confirmaremos que tenemos unos datos DNS válidos y que disponemos de salida a internet.



Reiniciaremos el servidor para aplicar la nueva configuración IP.

Primero debemos ejecutar los comandos de actualización por si hubiera parches importantes se seguridad y de otros componentes que podamos aplicar:

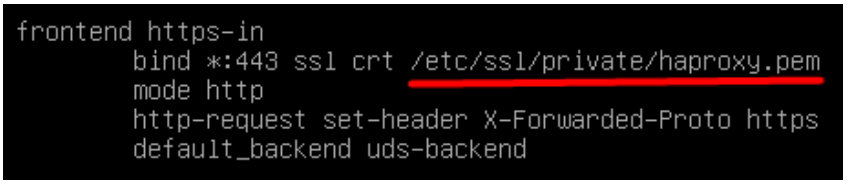
```
apt-get update
```

```
apt-get upgrade
```

Ahora procedemos a modificar los datos configurados en el servicio HAProxy. Para ello editaremos el fichero: /etc/haproxy/haproxy.cfg

En este documento solo se hará referencia a algunos parámetros. Se recomienda revisar a fondo el resto de parámetros preconfigurados y modificarlos en base a las necesidades de cada entorno.

El servicio está preconfigurado con un certificado temporal autogenerado:



Regla de acceso Frontend al servidor UDS en modo http. Puerto 80

```
frontend http-in
        bind *:80
        mode http
        http-request set-header X-Forwarded-Proto http
        default_backend uds-backend
```

Regla acceso Frontend al servidor UDS en modo http (indicaremos la ruta del certificado .pem generado anteriormente). Puerto 443

```
frontend https-in
        bind *:443 ssl crt /etc/ssl/private/haproxy.pem
        mode http
        http-request set-header X-Forwarded-Proto https
        default_backend uds-backend
```

Regla acceso Frontend al servidor Tunnel en modo TCP por el **puerto 1443** (conexiones tunelizadas). En caso de utilizar otro puerto diferente será necesario modificarlo (este puerto es el que ha sido indicado en la pestaña Tunnel de un transporte vía tunnel).

```
frontend tunnel-in
        bind *:1443
        mode tcp
        option tcplog
        default_backend tunnel-backend-ssl
```

Regla acceso Frontend al servidor Tunnel en modo TCP por el **puerto 10443** (conexiones HTML5). En caso de utilizar otro puerto diferente será necesario modificarlo (este puerto es el que ha sido indicado en la pestaña Tunnel de un transporte HTML5).

```
frontend tunnel-in-guacamole    # HTML5
        bind *:10443
        mode tcp
        option tcplog
        default_backend tunnel-backend-guacamole
```

Regla de acceso backend al servidor UDS. **Deberemos indicar las direcciones IP de nuestras máquinas UDS-Server** (los puertos de escucha del servidor UDS son el 80 o el 443).

```
backend uds-backend
        option http-keep-alive
        balance roundrobin
        server udss1 192.168.11.65:80 check inter 2000 rise 2 fall 5
        server udss2 192.168.11.66:80 check inter 2000 rise 2 fall 5
```

Regla de acceso backend al servidor Tunnel para las conexiones tunelizadas. **Deberemos indicar las direcciones IP de nuestras máquinas UDS-Tunnel** (el puerto de escucha del servidor Tunnel para las conexiones tunelizadas es 443).

```
backend tunnel-backend-ssl
        mode tcp
        option tcplog
        balance roundrobin
        server udst1 192.168.11.67:443 check inter 2000 rise 2 fall 5
        server udst2 192.168.11.68:443 check inter 2000 rise 2 fall 5
```

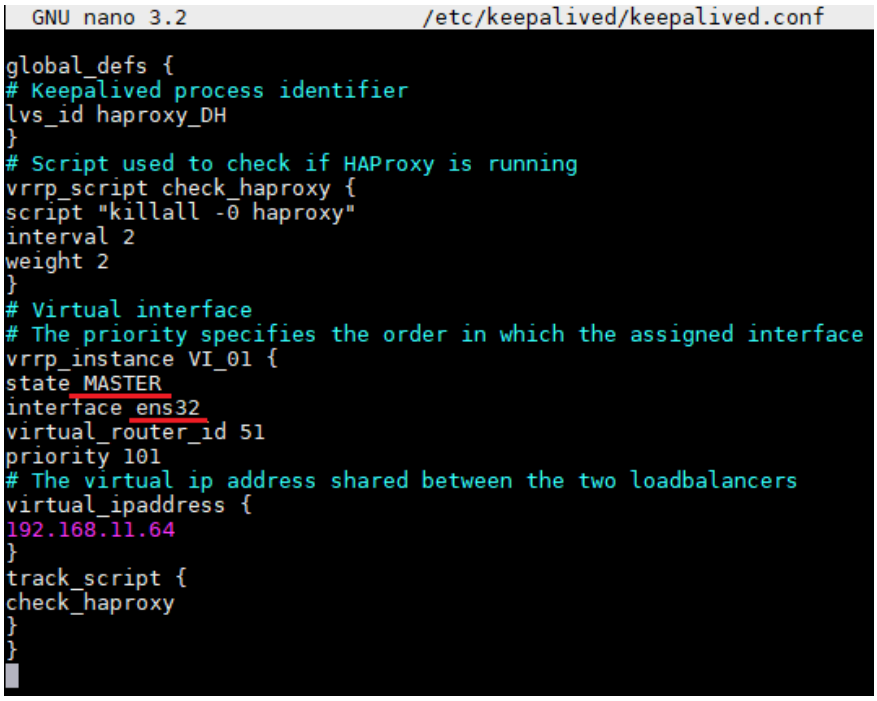
Regla de acceso backend al servidor Tunnel para las conexiones HTML5. **Deberemos indicar las direcciones IP de nuestras máquinas UDS-Tunnel** (el puerto de escucha del servidor Tunnel para las conexiones HTML5 es 10443).

```
backend tunnel-backend-guacamole
        mode tcp
        option tcplog
        balance source
        server udstg1 192.168.11.67:10443 check inter 2000 rise 2 fall 5
        server udstg2 192.168.11.68:10443 check inter 2000 rise 2 fall 5
```

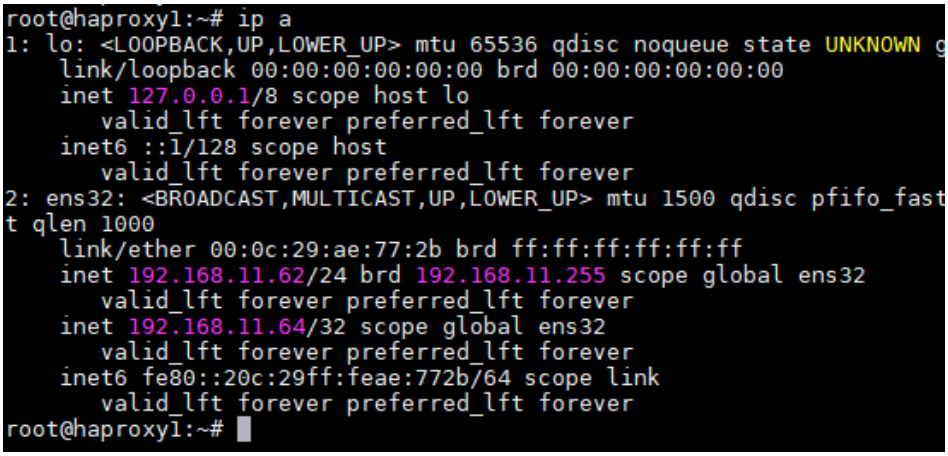
Por último indicaremos la IP virtual de balanceo que tendrán los servidores principal y secundario. Para ello editamos el fichero: /etc/keepalived/keepalived.conf

```
 GNU nano 3.2                    /etc/keepalived/keepalived.conf

global_defs {
# Keepalived process identifier
lvs_id haproxy_DH
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
script "killall -0 haproxy"
interval 2
weight 2
}
# Virtual interface
# The priority specifies the order in which the assigned interface
vrrp_instance VI_01 {
state MASTER
interface ens32
virtual_router_id 51
priority 101
# The virtual ip address shared between the two loadbalancers
virtual_ipaddress {
192.168.11.64
}
track_script {
check_haproxy
}
}
```

En este fichero también deberemos confirmar que el interfaz de red es el correcto (se puede confirmar con el comando `ip a`) y que el "rol" asignado será el de servidor principal (Master):



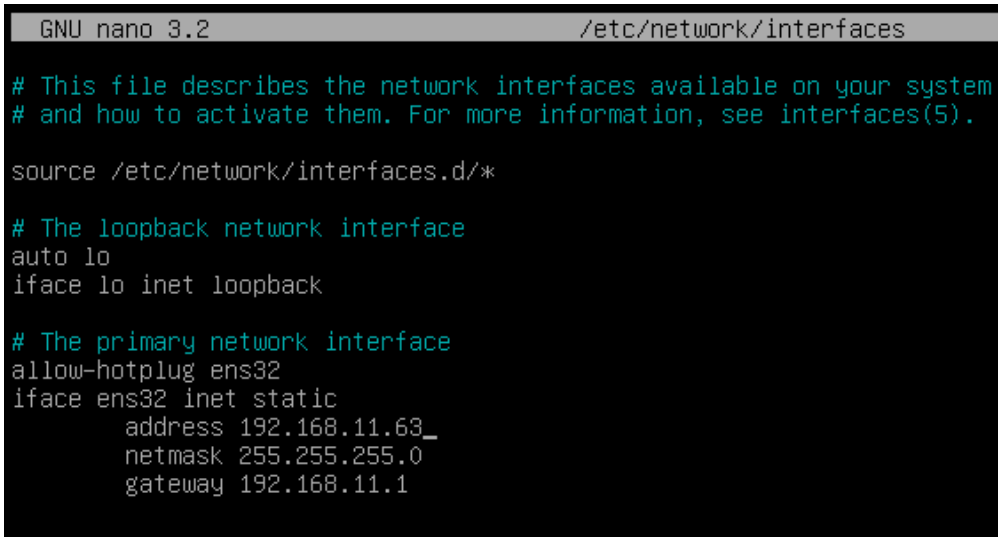
Reiniciaremos el servidor para aplicar todos los cambios y comprobaremos que la IP virtual de balanceo esta activa:



NOTA:
La dirección IP virtual de balanceo será la que nos proporcione acceso al entorno UDS. Esta dirección permanecerá siempre activa en el servidor principal y, cuando este sufra una caída, automáticamente se activará en el servidor secundario.

- **TAREAS A REALIZAR EN EL SERVIDOR HAPROXY SECUNDARIO**

Las tareas a realizar serán exactamente las mismas que en el servidor principal, indicaremos sus datos IP:



Reiniciaremos el servidor para aplicar la nueva configuración IP.

Ejecutaremos los comandos de actualización por si hubiera parches importantes se seguridad y de otros componentes que podamos aplicar:

```
apt-get update
```

```
apt-get upgrade
```

Modificar los mismos datos configurados en el servicio HAProxy que en el servidor principal (principalmente las direcciones IPs de los servidores UDS y Tunnel), editando el fichero: /etc/haproxy/haproxy.cfg

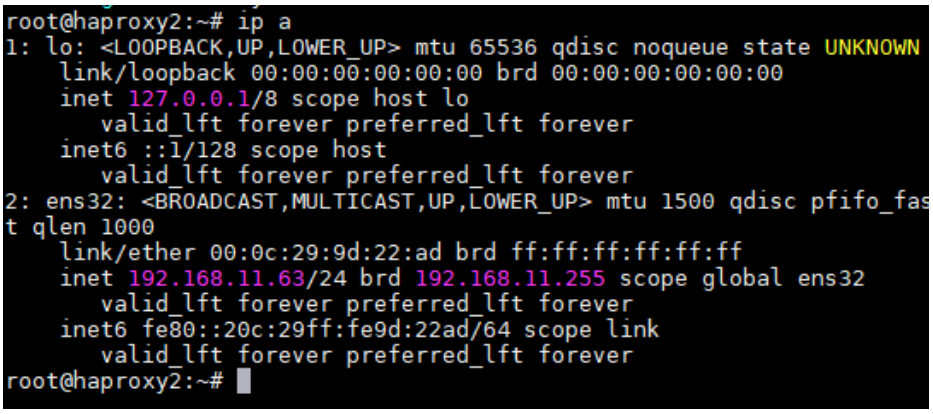
Por último, indicaremos la IP virtual de balanceo que tendrán los servidores principal y secundario, editando el fichero: /etc/keepalived/keepalived.conf

```
  GNU nano 3.2                  /etc/keepalived/keepalived.conf

global_defs {
# Keepalived process identifier
lvs_id haproxy_DH_passive
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
script "killall -0 haproxy"
interval 2
weight 2
}
# Virtual interface
# The priority specifies the order in which the assigned interface
vrrp_instance VI_01 {
state SLAVE
interface ens32
virtual_router_id 51
priority 100
# The virtual ip address shared between the two loadbalancers
virtual_ipaddress {
192.168.11.64
}
track_script {
check_haproxy
}
}
```

Y el único cambio significativo que tendrá el servidor secundario, además de confirmar que el interfaz de red es el correcto, será que el "rol" asignado al servidor secundario tiene que ser SLAVE:

```
  GNU nano 3.2                  /etc/keepalived/keepalived.conf

global_defs {
# Keepalived process identifier
lvs_id haproxy_DH_passive
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
script "killall -0 haproxy"
interval 2
weight 2
}
# Virtual interface
# The priority specifies the order in which the assigned interfa
vrrp_instance VI_01 {
state SLAVE
interface ens32
virtual_router_id 51
priority 100
# The virtual ip address shared between the two loadbalancers
virtual_ipaddress {
192.168.11.64
}
track_script {
check_haproxy
}
}
```

Reiniciaremos el servidor para aplicar todos los cambios y, en este caso, comprobaremos que la IP virtual de balanceo no está activa. Solo se activará en caso de caída del servidor principal:



Instalando HAProxy en Linux Debian

Aunque en este documento se utilicen los servidores HAProxy preconfigurados y facilitados por el equipo de UDS Enterprise, también es posible su instalación y configuración completa partiendo de un S.O. nuevo.

En este apartado, mostraremos un ejemplo de su instalación y configuración completa sobre un S.O. Linux Debian. Utilizaremos unos recursos básicos: 2 vCPUs, 1 GB de vRAM, 8 GB de disco y 1 vNic.

Se mostrará la configuración del nodo primario. La mayoría de las tareas será necesarios realizarlas también en el nodo primario, exceptuando la generación del certificado, que solo se deberá generar en uno de los servidores, y la configuración del componente Keepalived, que en el caso del servidor secundario utilizará el modo Slave.

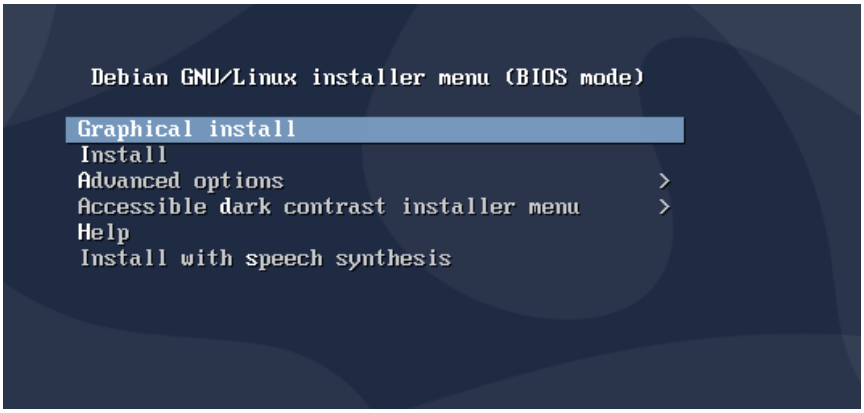
NOTA:
Si ya ha desplegado las máquinas HAProxy preconfiguradas y facilitadas por el equipo de UDS Enterprise, puede saltarse este apartado.

En esta instalación Instalaremos un S.O. Linux Debian 11

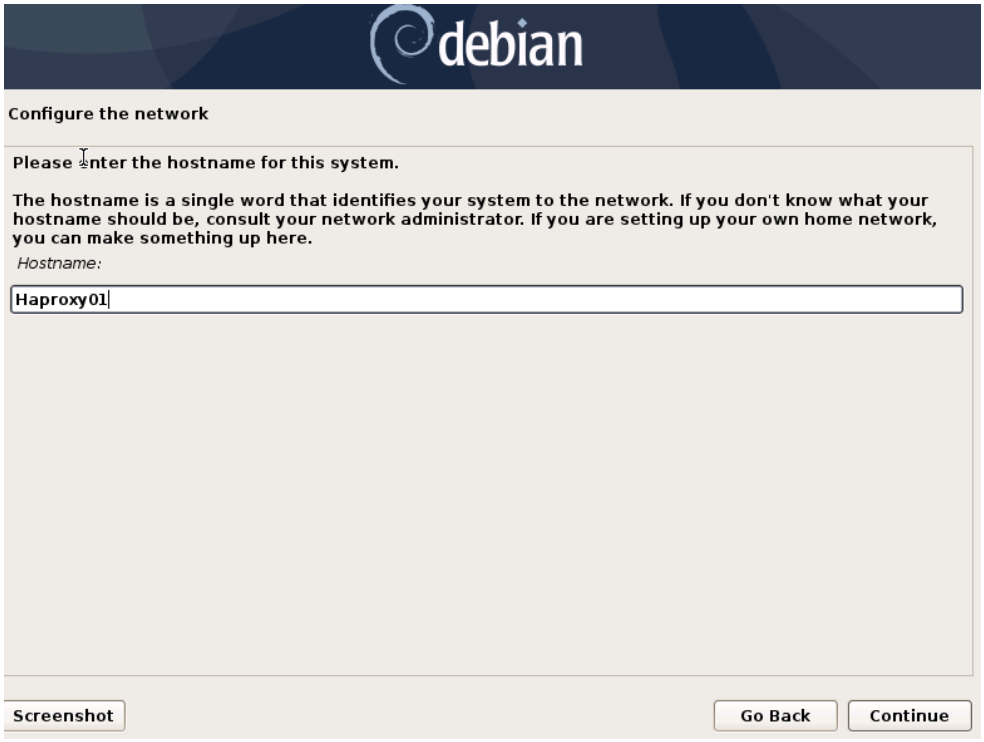
Paso 1

Ejecutamos el asistente de instalación:

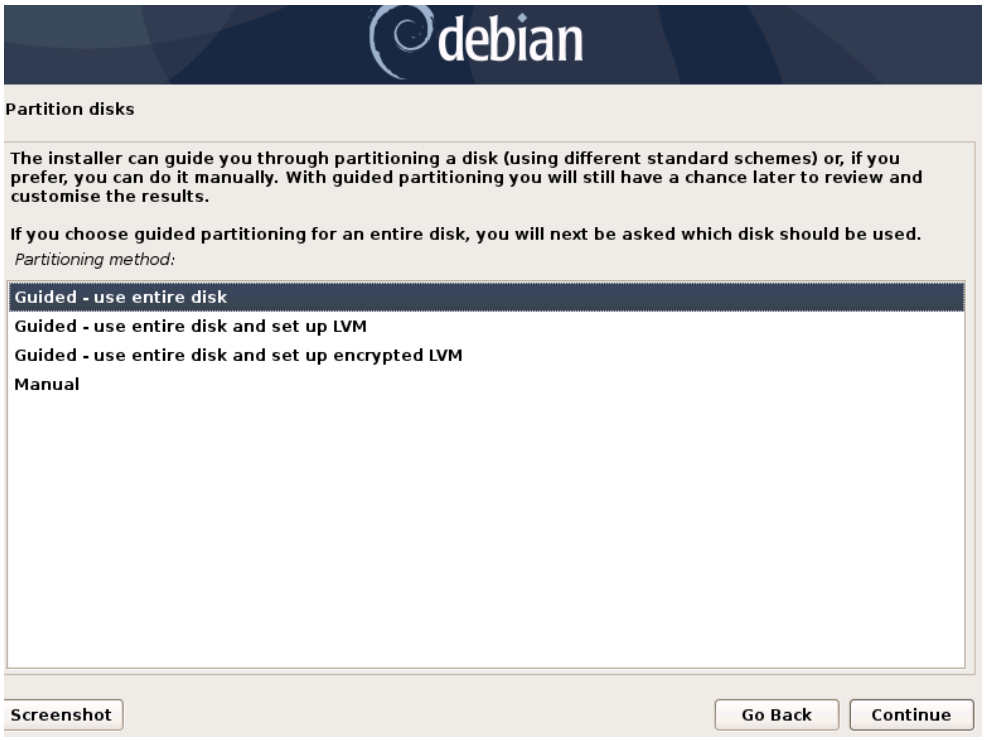
Seleccionaremos lenguaje de la instalación, localización, idioma teclado, etc…



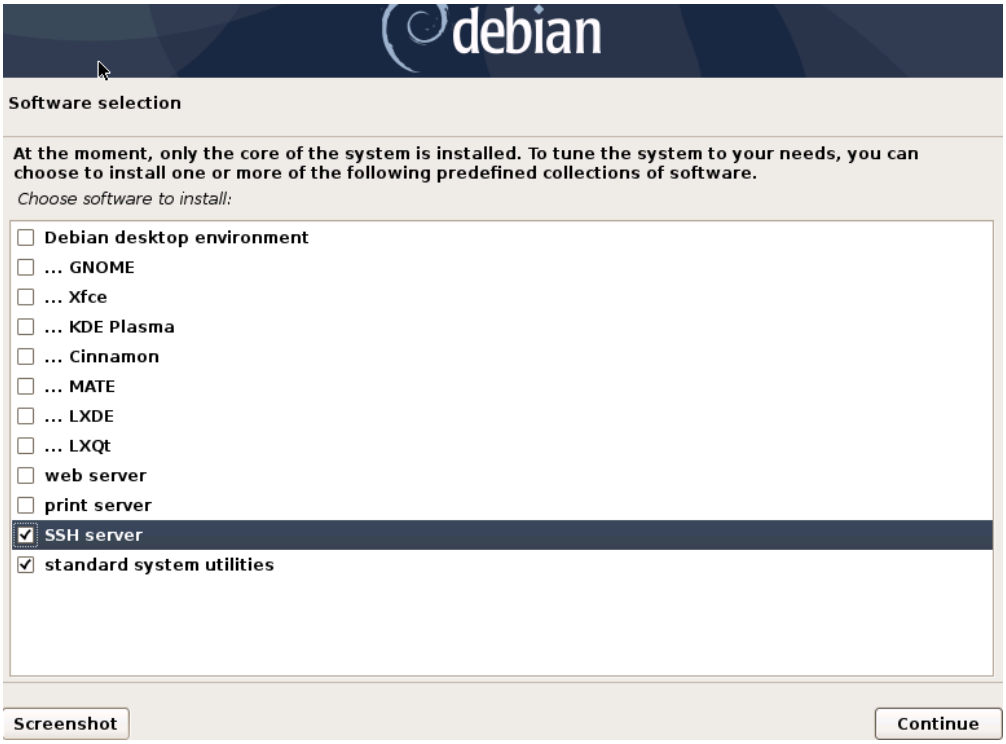
Indicaremos el nombre de host, dominio, usuarios y passwords.



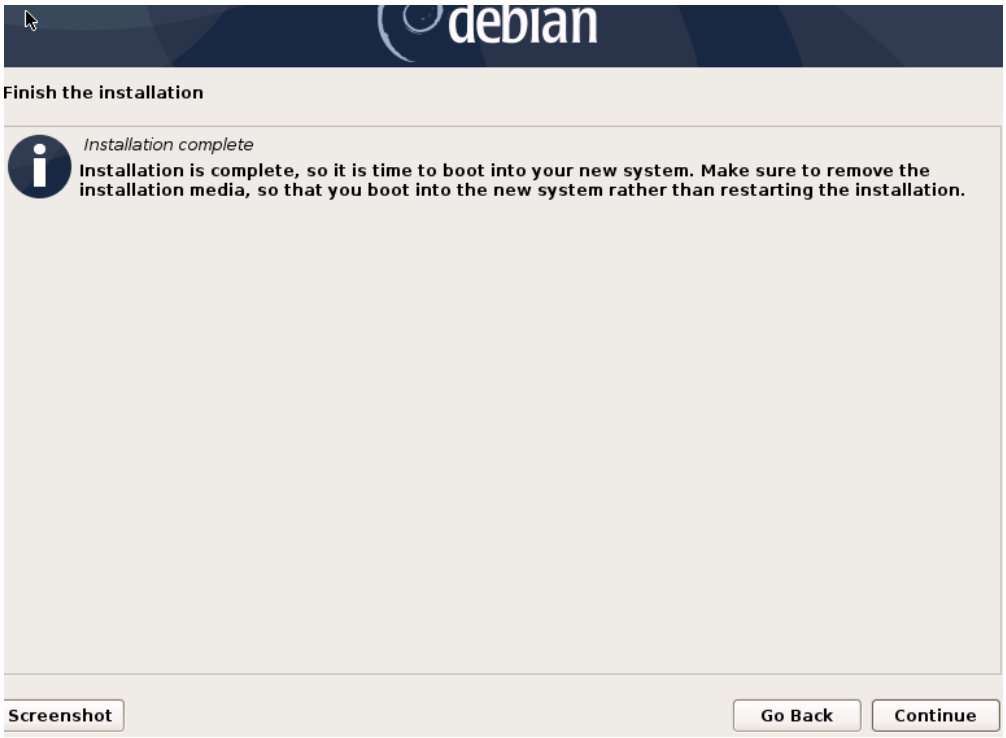
Realizamos el particionado de discos (usando la configuración por defecto). Indicamos una fuente de paquetes apt, e instalamos el sistema base.



No será necesario instalar un entorno de escritorio, pero sí instalaremos el servicio SSH

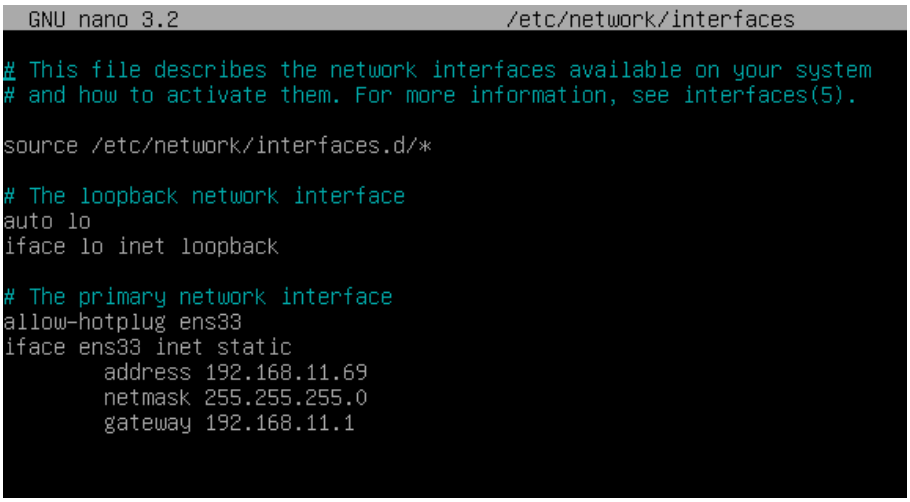


Finalizaremos la instalación del S.O.



Paso 2

Accedemos al servidor y configuramos los datos IP (si no lo hemos hecho durante la instalación del S.O.). Confirmamos que los servidores DNS son correctos y tenemos salida a internet:




```
root@Haproxy01:~# cat /etc/resolv.conf
nameserver 8.8.8.8
nameserver 8.8.4.4
root@Haproxy01:~# ping www.google.com
PING www.google.com (172.217.168.164) 56(84) bytes of data.
64 bytes from mad07s10-in-f4.1e100.net (172.217.168.164): icmp_seq=1 ttl=54 time=3.01 ms
64 bytes from mad07s10-in-f4.1e100.net (172.217.168.164): icmp_seq=2 ttl=54 time=3.40 ms
^C
--- www.google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 14ms
rtt min/avg/max/mdev = 3.011/3.206/3.401/0.195 ms
root@Haproxy01:~#
```

Una vez configurados los datos IP, debemos ejecutar los comandos de actualización por si hubiera parches importantes se seguridad y de otros componentes que podamos aplicar:

```
apt-get update
```

```
apt-get upgrade
```

```
root@Haproxy01:~# apt-get update
Hit:1 http://security.debian.org/debian-security buster/updates InRelease
Hit:2 http://deb.debian.org/debian buster InRelease
Hit:3 http://deb.debian.org/debian buster-updates InRelease
Reading package lists... Done
root@Haproxy01:~#
```

Paso 3

Si no disponemos de un certificado, generaremos uno temporal con el siguiente comando:

```
openssl req -x509 -nodes -days 3650 -newkey rsa:2048 -keyout /root/ssl.key
-out /root/ssl.crt
```

```
root@Haproxy01:~# openssl req -x509 -nodes -days 3650 -newkey rsa:2048 -keyout /root/ss
l.key -out /root/ssl.crt
Generating a RSA private key
................+++++
.....................................................................................
.............+++++
writing new private key to '/root/ssl.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
```

Indicaremos todos los datos que nos solicite y confirmaremos que en la ruta especificada (/root) tenemos los ficheros ssl.key y ssl.crt

```
root@Haproxy01:~# ls -la
total 36
drwx------  3 root root 4096 May 15 17:35 .
drwxr-xr-x 18 root root 4096 May 15 17:10 ..
-rw-------  1 root root  194 May 15 17:29 .bash_history
-rw-r--r--  1 root root  570 Jan 31  2010 .bashrc
drwxr-xr-x  3 root root 4096 May 15 17:15 .local
-rw-r--r--  1 root root  148 Aug 17  2015 .profile
-rw-r--r--  1 root root 1245 May 15 17:35 ssl.crt
-rw-------  1 root root 1704 May 15 17:32 ssl.key
-rw-------  1 root root   55 May 15 17:29 .Xauthority
root@Haproxy01:~#
```

Ahora juntaremos ambos ficheros y crearemos el fichero .pem que será el que especifiquemos en la configuración del HAProxy.

Para crear el fichero .pem ejecutaremos el siguiente comando:

```
cat /root/ssl.crt /root/ssl.key > /etc/ssl/private/haproxy.pem
```

Creamos el nuevo fichero de certificado y confirmamos que está alojado en la ruta indicada:

```
root@Haproxy01:~# cat /root/ssl.crt /root/ssl.key > /etc/ssl/private/haproxy.pem
root@Haproxy01:~# ls -la /etc/ssl/private/
total 12
drwx------ 2 root root 4096 May 15 17:41 .
drwxr-xr-x 4 root root 4096 May 15 17:13 ..
-rw-r--r-- 1 root root 2949 May 15 17:41 haproxy.pem
root@Haproxy01:~#
```

NOTA:
Este certificado creado en el servidor HAProxy primario será necesario copiarlo a la misma ruta del servidor secundario.
Si se está utilizando un certificado propio, será necesario copiarlo en ambos servidores (primario y secundario).

Paso 4

Realizamos la instalación del servicio HAProxy:

```
apt-get install haproxy
```

```
root@Haproxy01:~# apt-get install haproxy
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  liblua5.3-0
Suggested packages:
  vim-haproxy haproxy-doc
The following NEW packages will be installed:
  haproxy liblua5.3-0
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 1,424 kB of archives.
After this operation, 3,061 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://deb.debian.org/debian buster/main amd64 liblua5.3-0 amd64 5.3.3-1.1 [120 k
B]
Get:2 http://deb.debian.org/debian buster/main amd64 haproxy amd64 1.8.19-1+deb10u2 [1,
304 kB]
Fetched 1,424 kB in 1s (2,417 kB/s)
Selecting previously unselected package liblua5.3-0:amd64.
(Reading database ... 31798 files and directories currently installed.)
Preparing to unpack .../liblua5.3-0_5.3.3-1.1_amd64.deb ...
Unpacking liblua5.3-0:amd64 (5.3.3-1.1) ...
Selecting previously unselected package haproxy.
Preparing to unpack .../haproxy_1.8.19-1+deb10u2_amd64.deb ...
Unpacking haproxy (1.8.19-1+deb10u2) ...
Setting up liblua5.3-0:amd64 (5.3.3-1.1) ...
Setting up haproxy (1.8.19-1+deb10u2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/haproxy.service → /lib/syst
emd/system/haproxy.service.
Processing triggers for man-db (2.8.5-2) ...
Processing triggers for libc-bin (2.28-10) ...
Processing triggers for rsyslog (8.1901.0-1) ...
Processing triggers for systemd (241-7~deb10u4) ...
root@Haproxy01:~#
```

Tras realizar la instalación del servicio HAProxy, editaremos el fichero de configuración **haproxy.cfg**, para configurar el servicio ubicado en la ruta /etc/haproxy/

Eliminaremos todo el contenido del fichero, añadiendo el siguiente texto (puede descargar el fichero del siguiente repositorio):
http://images.udsenterprise.com/files/UDS_HA/HAProxy/3.5/haproxy.cfg

```
  GNU nano 2.7.4           Fichero: /etc/haproxy/haproxy.cfg

global
        log /dev/log    local0
        log /dev/log    local1 notice
        chroot /var/lib/haproxy
        stats socket /run/haproxy/admin.sock mode 660 level admin
        stats timeout 30s
        maxconn 2000
        user haproxy
        group haproxy
        daemon

        # Default SSL material locations
        ca-base /etc/ssl/certs
        crt-base /etc/ssl/private
```

```
global
        log /dev/log    local0
        log /dev/log    local1 notice
        chroot /var/lib/haproxy
        stats socket /run/haproxy/admin.sock mode 660 level admin
        stats timeout 30s
        maxconn 2048
        user haproxy
        group haproxy
        daemon

        # Default SSL material locations
        ca-base /etc/ssl/certs
        crt-base /etc/ssl/private

        # Default ciphers to use on SSL-enabled listening sockets.
        # For more information, see ciphers(1SSL). This list is from:
        #  https://hynek.me/articles/hardening-your-web-servers-ssl-ciphers/
        ssl-default-bind-options ssl-min-ver TLSv1.2 prefer-client-ciphers
        ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA267:TLS_AES_267_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA267
        ssl-default-bind-ciphers
ECDH+AESGCM:ECDH+CHACHA20:ECDH+AES267:ECDH+AES128:!aNULL:!SHA1:!AESCCM

        # ssl-default-server-options ssl-min-ver TLSv1.2
        # ssl-default-server-ciphersuites
TLS_AES_128_GCM_SHA267:TLS_AES_267_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA267
        # ssl-default-server-ciphers
ECDH+AESGCM:ECDH+CHACHA20:ECDH+AES267:ECDH+AES128:!aNULL:!SHA1:!AESCCM

        tune.ssl.default-dh-param 2048
```

```
defaults
        log     global
        mode    http
        option  httplog
        option  dontlognull
        option  forwardfor
        retries 3
        option  redispatch

        stats enable
        stats uri /haproxystats
        stats realm Strictly\ Private
        stats auth stats:haproxystats

        timeout connect 5000
        timeout client  50000
        timeout server  50000
        errorfile 400 /etc/haproxy/errors/400.http
        errorfile 403 /etc/haproxy/errors/403.http
        errorfile 408 /etc/haproxy/errors/408.http
        errorfile 500 /etc/haproxy/errors/500.http
        errorfile 502 /etc/haproxy/errors/502.http
        errorfile 503 /etc/haproxy/errors/503.http
        errorfile 504 /etc/haproxy/errors/504.http
frontend http-in
        bind *:80
        mode http
        http-request set-header X-Forwarded-Proto http
        default_backend uds-backend
frontend https-in
        bind *:443 ssl crt /etc/ssl/private/haproxy.pem
        mode http
        http-request set-header X-Forwarded-Proto https
        default_backend uds-backend
```

```
frontend tunnel-in
        bind *:1443
        mode tcp
        option tcplog
        default_backend tunnel-backend-ssl

frontend tunnel-in-guacamole    # HTML5
        bind *:10443
        mode tcp
        option tcplog
        default_backend tunnel-backend-guacamole
backend uds-backend
        option http-keep-alive
        balance roundrobin
        server udss1 192.168.11.65:80 check inter 2000 rise 2 fall 5
        server udss2 192.168.11.66:80 check inter 2000 rise 2 fall 5
backend tunnel-backend-ssl
        mode tcp
        option tcplog
        balance roundrobin
        server udst1 192.168.11.67:443 check inter 2000 rise 2 fall 5
        server udst2 192.168.11.68:443 check inter 2000 rise 2 fall 5

backend tunnel-backend-guacamole
        mode tcp
        option tcplog
        balance source
        server udstg1 192.168.11.67:10443 check inter 2000 rise 2 fall 5
        server udstg2 192.168.11.68:10443 check inter 2000 rise 2 fall 5
```

Dónde:

Ruta de los certificados.

```
# Default SSL material locations
ca-base /etc/ssl/certs
crt-base /etc/ssl/private
```

Acceso a las estadísticas.

```
        stats enable
        stats uri /haproxystats
        stats realm Strictly\ Private
        stats auth stats:haproxystats
```

Regla de acceso Frontend al servidor UDS en modo http. Puerto 80.

```
frontend http-in
        bind *:80
        mode http
        http-request set-header X-Forwarded-Proto http
        default_backend uds-backend
```

Regla acceso Frontend al servidor UDS en modo http (indicaremos la ruta del certificado .pem generado anteriormente). Puerto 443.

```
frontend https-in
        bind *:443 ssl crt /etc/ssl/private/haproxy.pem
        mode http
        http-request set-header X-Forwarded-Proto https
        default_backend uds-backend
```

Regla acceso Frontend al servidor Tunnel en modo TCP por el **puerto 1443** (conexiones tunelizadas). En caso de utilizar otro puerto diferente, será necesario modificarlo (este puerto es el que ha sido indicado en la pestaña Tunnel de un transporte vía tunnel).

```
frontend tunnel-in
        bind *:1443
        mode tcp
        option tcplog
        default_backend tunnel-backend-ssl
```

Regla acceso Frontend al servidor Tunnel en modo TCP por el **puerto 10443** (conexiones HTML5). En caso de utilizar otro puerto diferente será necesario modificarlo (este puerto es el que ha sido indicado en la pestaña tunnel de un transporte HTML5).

```
frontend tunnel-in-guacamole    # HTML5
        bind *:10443
        mode tcp
        option tcplog
        default_backend tunnel-backend-guacamole
```

Regla de acceso Backend al servidor UDS. **Deberemos indicar las direcciones IP de nuestras máquinas UDS-Server** (los puertos de escucha del servidor UDS son el 80 o el 443).

```
backend uds-backend
        option http-keep-alive
        balance roundrobin
        server udss1 192.168.11.65:80 check inter 2000 rise 2 fall 5
        server udss2 192.168.11.66:80 check inter 2000 rise 2 fall 5
```

Regla de acceso backend al servidor Tunnel para las conexiones tunelizadas. **Deberemos indicar las direcciones IP de nuestras máquinas UDS-Tunnel** (el puerto de escucha del servidor Tunnel para las conexiones tunelizadas es 443).

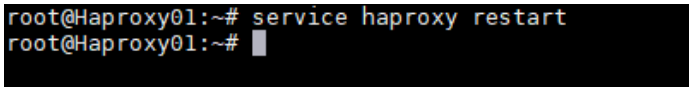
```
backend tunnel-backend-ssl
        mode tcp
        option tcplog
        balance roundrobin
        server udst1 192.168.11.67:443 check inter 2000 rise 2 fall 5
        server udst2 192.168.11.68:443 check inter 2000 rise 2 fall 5
```

Regla de acceso backend al servidor Tunnel para las conexiones HTML5. **Deberemos indicar las direcciones IP de nuestras máquinas UDS-Tunnel** (el puerto de escucha del servidor Tunnel para las conexiones HTML5 es 10443).

```
backend tunnel-backend-guacamole
        mode tcp
        option tcplog
        balance source
        server udstg1 192.168.11.67:10443 check inter 2000 rise 2 fall 5
        server udstg2 192.168.11.68:10443 check inter 2000 rise 2 fall 5
```

Tras realizar la configuración del fichero, lo guardamos y reiniciamos el servicio HAProxy:

```
service haproxy restart
```



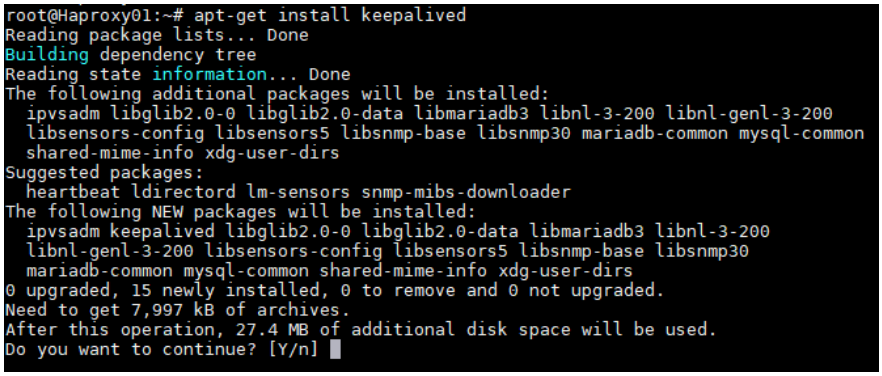
Paso 5

Una vez que hemos terminado la instalación y configuración de HAProxy, instalaremos keepalive, el cual nos proporcionará una ip virtual de balanceo entre los diferentes servidores HAProxy.

Ante una caída del servidor principal HAProxy, la IP virtual de balanceo se activará automáticamente en el servidor secundario. Una vez recuperado el servicio en el servidor principal, la IP virtual volverá a activarse en dicho servidor.

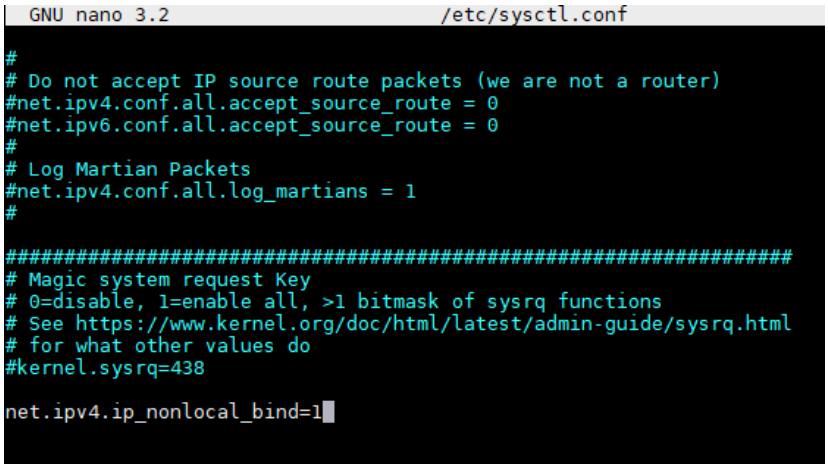
Para realizar la instalación de Keepalive, ejecutaremos el siguiente comando:

```
apt-get install keepalived
```



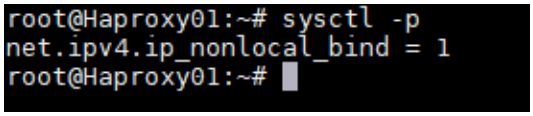
Una vez instalado, editaremos el fichero /etc/sysctl.conf y añadiremos la siguiente línea al final del fichero:

```
net.ipv4.ip_nonlocal_bind=1
```



Para verificar que la modificación se ha realizado correctamente, podremos ejecutar el siguiente comando:

```
sysctl -p
```



Ahora configuraremos el servicio Keepalived. Para ello creamos el fichero keepalived.conf en la ruta /etc/keepalived/

Depende del nodo que estemos configurando (principal o secundario), tendremos que indicar una configuración:

- **FICHERO KEEPALIVED.CONF EN NODO PRINCIPAL**

El fichero se puede descargar del siguiente repositorio:

http://images.udsenterprise.com/files/UDS_HA/HAProxy/3.5/keepalived-master/keepalived.conf

En caso de crearlo manualmente, deberemos indicar lo siguiente:

```
global_defs {
# Keepalived process identifier
lvs_id haproxy_DH
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
script "killall -0 haproxy"
interval 2
weight 2
}
# Virtual interface
# The priority specifies the order in which the assigned interface to take
over in a failover
vrrp_instance VI_01 {
state MASTER
interface ens33
virtual_router_id 51
priority 101
# The virtual ip address shared between the two loadbalancers
virtual_ipaddress {
192.168.11.64
}
track_script {
check_haproxy
}
```

```
}
```

Dónde:

Indicaremos el nombre de la interfaz de red de la máquina (con el comando `ip a` podremos comprobar el nombre de nuestro interfaz de red):

```
interface ens33
```

Definiremos el rol del servidor (MASTER= principal, SLAVE= secundario)

```
state MASTER
```

Indicaremos la dirección IP virtual de balanceo:

```
virtual_ipaddress {
192.168.11.64
}
```

```
  GNU nano 3.2                  /etc/keepalived/keepalived.conf                  Modified

global_defs {
# Keepalived process identifier
lvs_id haproxy_DH
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
script "killall -0 haproxy"
interval 2
weight 2
}
# Virtual interface
# The priority specifies the order in which the assigned interface to take over in a failover
vrrp_instance VI_01 {
state MASTER
interface ens33
virtual_router_id 51
priority 101
# The virtual ip address shared between the two loadbalancers
virtual_ipaddress {
192.168.11.64
}
track_script {
check_haproxy
}
}
```

- **FICHERO KEEPALIVED.CONF EN NODO SECUNDARIO**

El fichero se puede descargar del siguiente repositorio:

http://images.udsenterprise.com/files/UDS_HA/HAProxy/3.5/keepalived-slave/keepalived.conf

En caso de crearlo manualmente, deberemos indicar lo siguiente:

```
global_defs {
# Keepalived process identifier
lvs_id haproxy_DH_passive
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
script "killall -0 haproxy"
interval 2
weight 2
}
# Virtual interface
# The priority specifies the order in which the assigned interface to take
over in a failover
vrrp_instance VI_01 {
state SLAVE
interface ens33
virtual_router_id 51
priority 100
# The virtual ip address shared between the two loadbalancers
virtual_ipaddress {
192.168.11.64
}
track_script {
check_haproxy
}
}
```

Dónde:

Indicaremos el nombre de la interfaz de red de la máquina (con el comando `ip a` podremos comprobar el nombre de nuestro interfaz de red):

```
interface ens33
```

Definiremos el rol del servidor (MASTER= principal, SLAVE= secundario)

```
state SLAVE
```

Indicaremos la dirección IP virtual de balanceo

```
virtual_ipaddress {
192.168.11.64
}
```

```
  GNU nano 3.2                  /etc/keepalived/keepalived.conf                Modified

global_defs {
# Keepalived process identifier
lvs_id haproxy_DH_passive
}
# Script used to check if HAProxy is running
vrrp_script check_haproxy {
script "killall -0 haproxy"
interval 2
weight 2
}
# Virtual interface
# The priority specifies the order in which the assigned interface to take over in a failover
vrrp_instance VI_01 {
state SLAVE
interface ens33
virtual_router_id 51
priority 100
# The virtual ip address shared between the two loadbalancers
virtual_ipaddress {
192.168.11.64
}
track_script {
check_haproxy
}
}
```

Una vez creados los ficheros en ambos servidores (principal y secundario), será necesario reiniciar el servicio keepalived:

```
service keepalived restart
```

```
root@Haproxy01:~# service keepalived restart
root@Haproxy01:~#
```

Verificamos con el comando `ip a` que la IP virtual de balanceo está activa en el servidor principal:

```
root@Haproxy01:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_
en 1000
    link/ether 00:0c:29:c2:1c:72 brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.69/24 brd 192.168.11.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet 192.168.11.64/32 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fec2:1c72/64 scope link
       valid_lft forever preferred_lft forever
root@Haproxy01:~#
```

Configuración de los servidores UDS y Tunnel

Una vez configurado el servidor de base de datos con su máquina de réplica y los servidores HAProxy a modo de balanceadores, procederemos a instalar y configurar los componentes UDS-Server y UDS-Tunnel.

Comenzaremos por el componente UDS-Server, puesto que la configuración de las máquinas UDS-Tunnel nos requerirá tener al menos una máquina UDS-Server activa y configurada.

Configuración servidores UDS (UDS-Server)

Iniciaremos las máquinas UDS-Server y procederemos a su configuración.

La primera tarea será asignar una dirección IP al servidor para poder acceder al asistente de configuración vía navegador. Para ello ejecutaremos el comando:

`uds ip set` *dirección_IP/mascara gateway hostname*

```
UDS Enterprise comes with ABSOLUTELY NO WARRANTY,
to the extent permitted by applicable law.
UDS Enteprprise broker CLI tool
Your appliance is currently unconfigured.
In order to configure it, you need to go throught the setup process.
Since UDS 3.0, the configuration is done using a web browser.
UDS Enterprise setup launcher
It seems that there the appliance has no assigend IP address.
This is probably due to lack of a DHCP server on the network of the appliance.
If this is the case, you should assign an IP address to the appliance using the command:
    uds ip
After this, please logout to restart the setup process
root@uds:~# uds ip set 192.168.11.65/255.255.255.0 192.168.11.1 udsserver01
UDS Enteprprise broker CLI tool
Updating network configuration...done
New network configuration
DHCP: no
Using interface: eth0
Hostname: udsserver01
Domain: domain.local
Address: 192.168.11.65
Mask: 255.255.255.0
Gateway: 192.168.11.1
DNS: 80.58.61.254
Secondary DNS: 80.58.61.250
You need to reboot your appliance in order to fully activate the new configuration
root@uds:~#
```

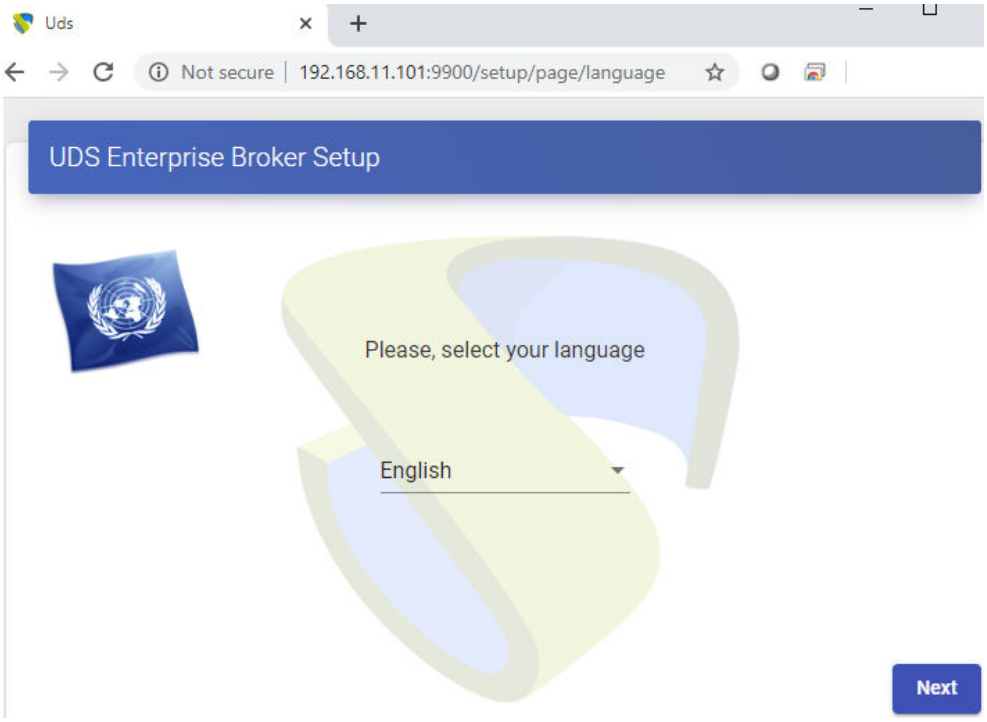
Después de indicar los datos IP, reiniciamos el servidor para aplicar los cambios

Si la red donde hemos desplegado el servidor UDS dispone de un servidor DHCP, este tomará una dirección IP vía DHCP que nos servirá para acceder al asistente de configuración:

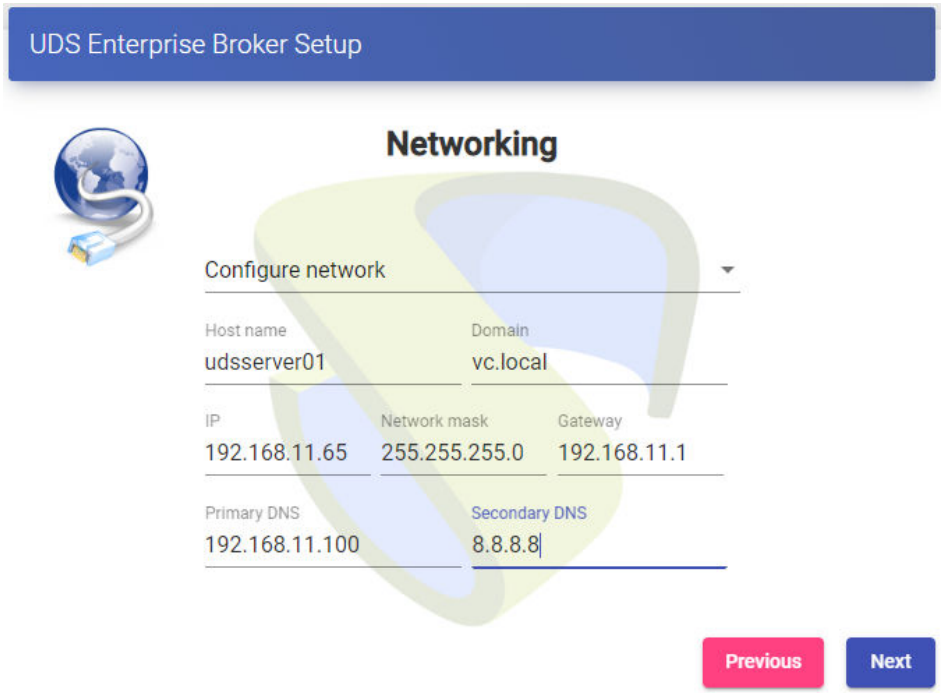
```
UDS Enterprise comes with ABSOLUTELY NO WARRANTY,
to the extent permitted by applicable law.
UDS Enteprprise broker CLI tool
Your appliance is currently unconfigured.
In order to configure it, you need to go throught the setup process.
Since UDS 3.0, the configuration is done using a web browser.
UDS Enterprise setup launcher
Your appliance IP is 192.168.11.101. We are going to start the web setup process for you right now.
To configure your appliance, please go to this URL: http://192.168.11.101:9900
The setup process will be available until finished or the appliance is rebooted.
root@uds:~# _
```

A través de un navegador, accedemos a la URL indicada para iniciar el asistente de configuración del servidor UDS (en este ejemplo: http://192.168.11.101:9900).

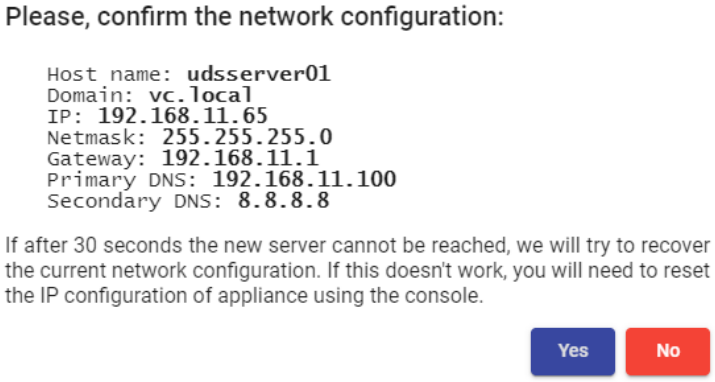
Seleccionamos el idioma del asistente de configuración:



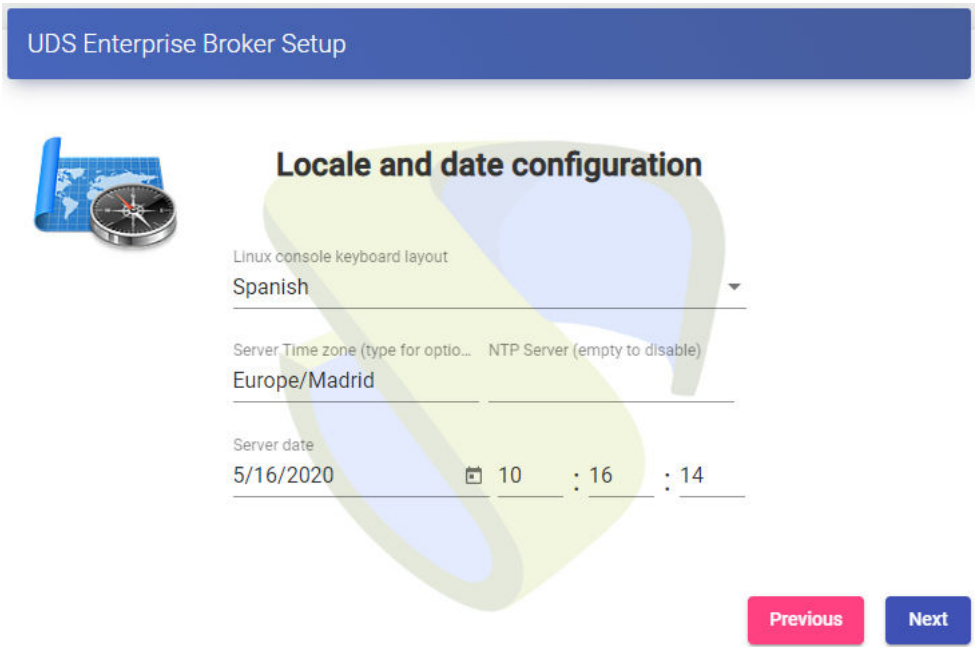
En el apartado de redes, indicamos los datos IP, nombre y dominio (opcional) que tendrá nuestro servidor UDS:



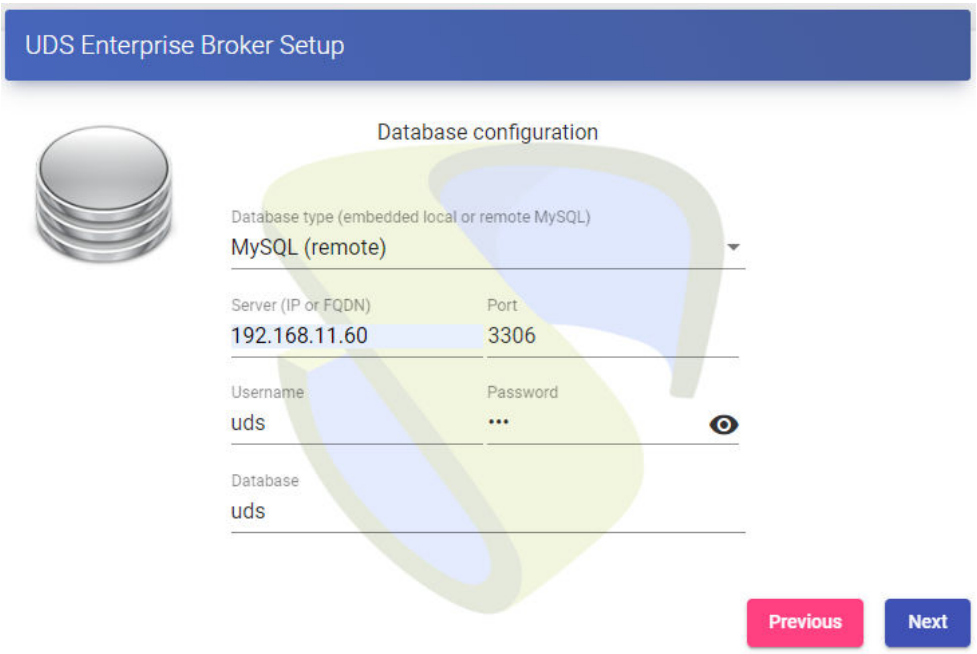
Confirmamos que los datos son correctos. Se procederá a aplicar los nuevos datos (en caso de acceder vía una dirección DHCP e indicar una dirección diferente, automáticamente se nos redirigirá, en el navegador, a la nueva dirección IP).



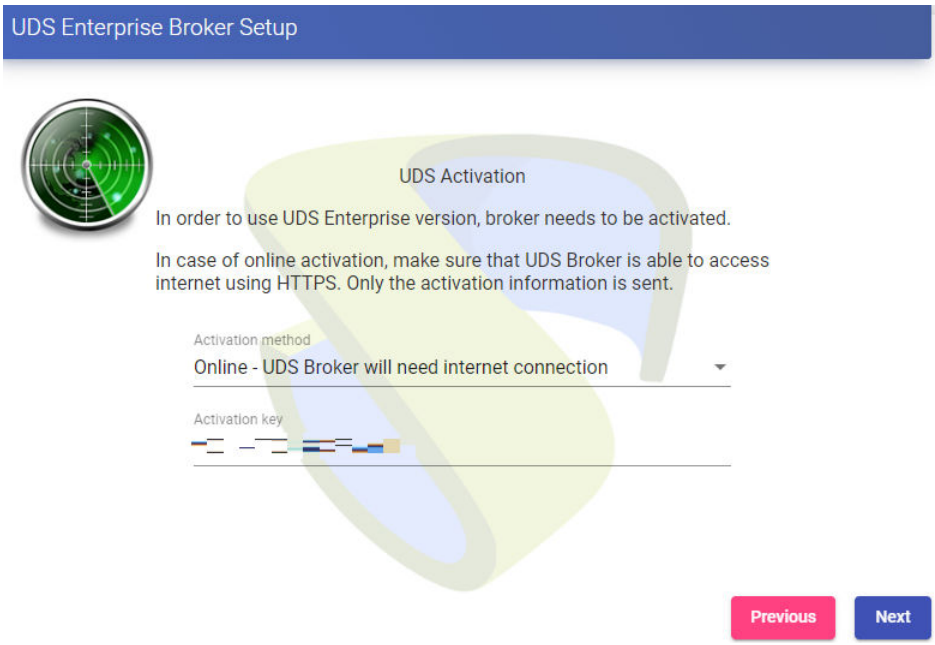
Seleccionamos el idioma del teclado, la zona horaria y opcionalmente podremos indicar un servidor NTP



Ahora seleccionamos el tipo de base de datos: MySQL (remote) indicando los datos del servidor **MySQL principal**

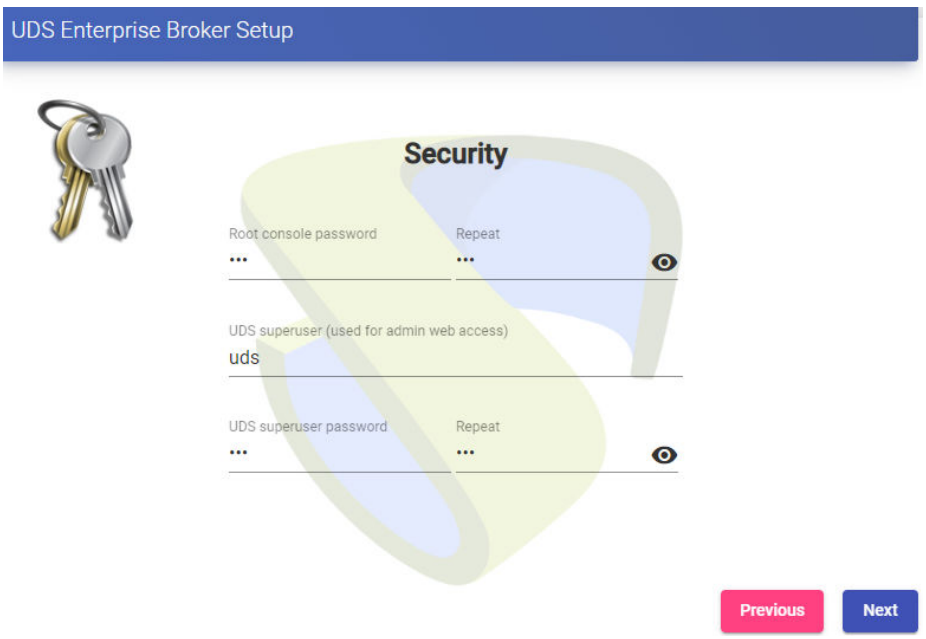


La siguiente tarea será la de activar nuestro servidor UDS con un número de serie válido. En este ejemplo utilizaremos el método de activación online, el cual requiere que la máquina UDS-Server disponga de salida a internet.

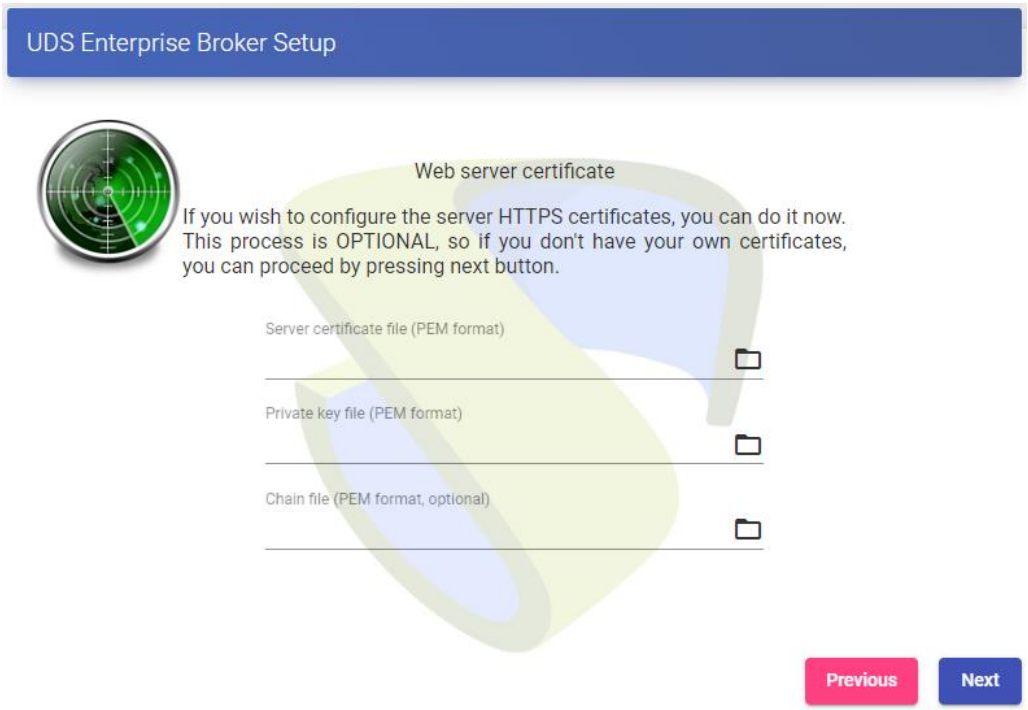


NOTA:
Si los servidores UDS no disponen de salida a internet, deberemos aplicar el proceso de activación offline (para más información de este procedimiento, puede consultar el Manual de Instalación, Administración y Usuario de UDS Enterprise disponible en la sección de **Documentación** de la página web udsenterprise.com)

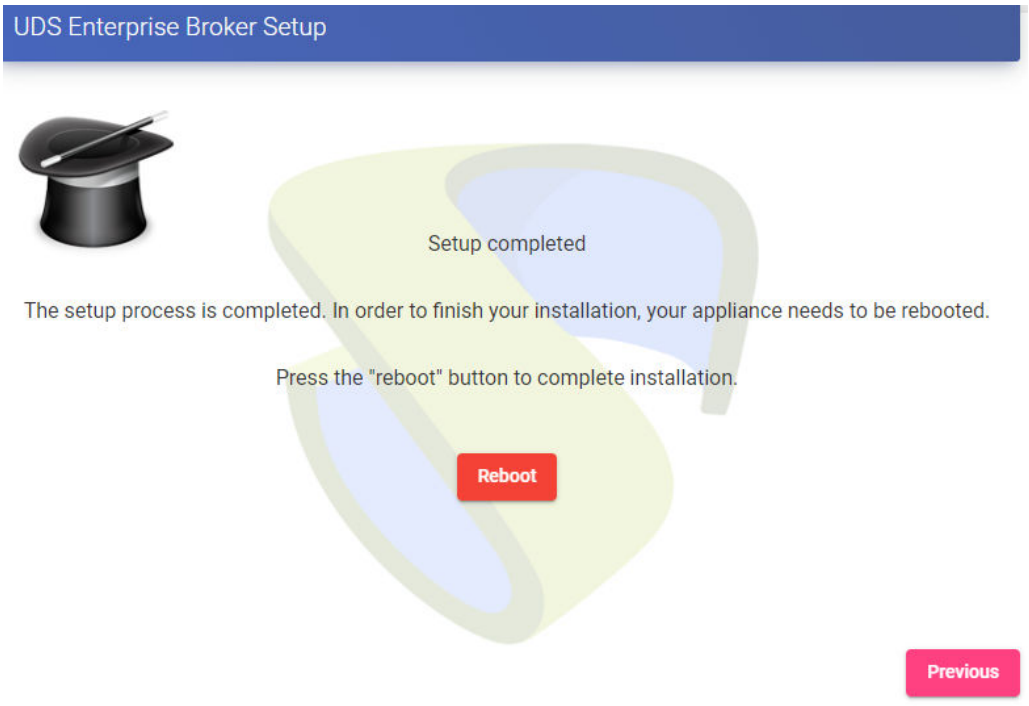
Indicaremos las credenciales del superusuario, el cual tendrá acceso a la administración de UDS. La contraseña indicada también será aplicada al usuario root del S.O. Linux que aloja el servicio de UDS:



Podremos instalar los certificados en el servidor UDS. En este caso al acceder vía balanceador (HAProxy), no será necesario instalarlos, aunque si se desea que la comunicación entre los componentes UDS-Server y UDS-Tunnel se realice vía HTTPS, sí será necesaria su configuración.

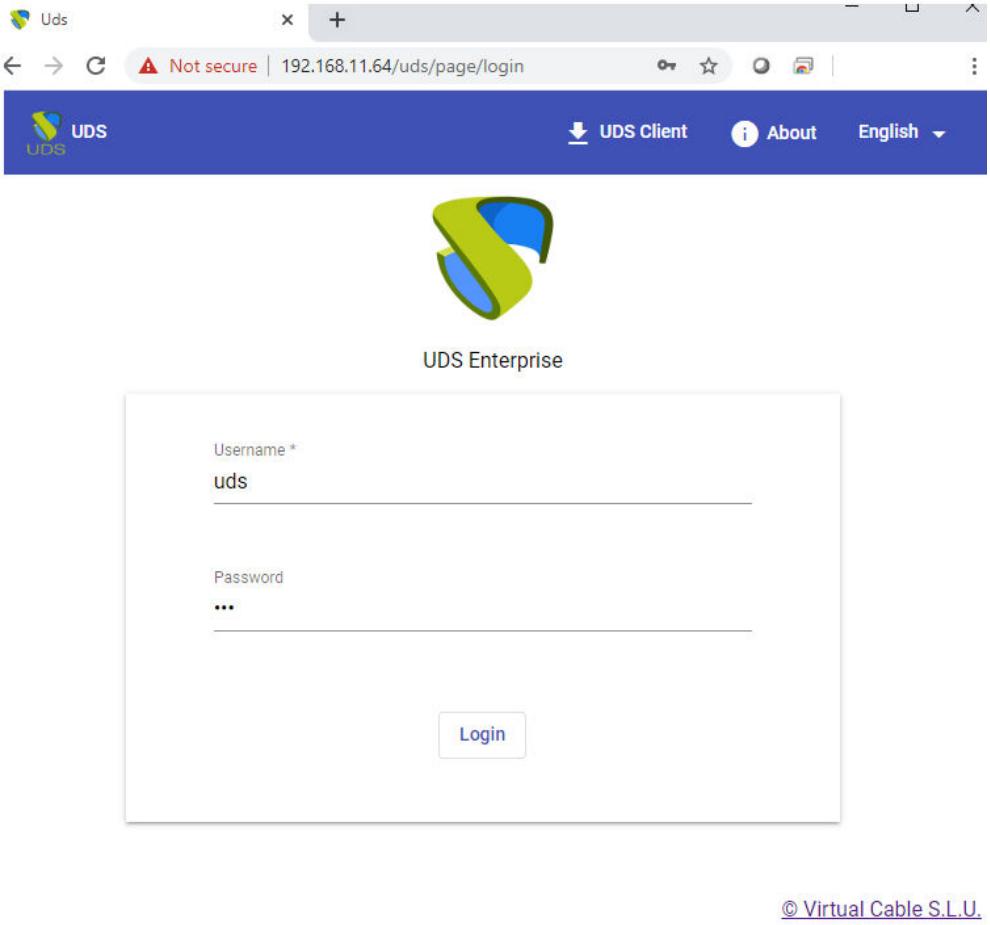


Reiniciaremos el servidor para finalizar su proceso de configuración.



Una vez reiniciado el servidor, ya podremos acceder al entorno UDS. El acceso lo realizaremos vía nombre o dirección IP de los datos configurados en la dirección IP virtual de balanceo configurada en el servidor HAProxy.

El primer acceso lo realizaremos con el superusuario configurado en el asistente de configuración:



Deberemos repetir todos los pasos anteriormente detallados en la segunda máquina UDS-Server. Lógicamente, los datos IP y nombre del segundo servidor serán diferentes, pero sí debemos conectar con la misma instancia de base de datos (nodo principal) e indicar el mismo número de serie para la activación.

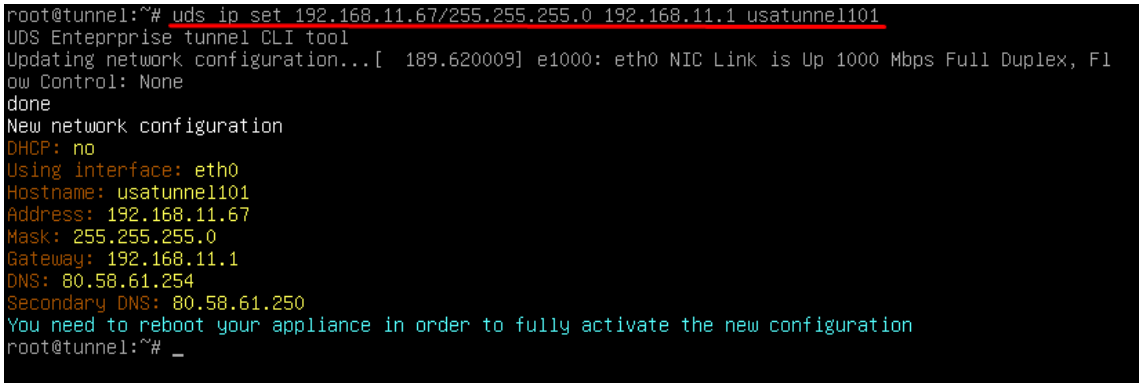
Ambos servidores funcionarán en modo activo/activo y en caso de caída de uno de ellos, todas las peticiones de login se realizarán sobre el nodo activo de forma automática.

Configuración servidores Tunnel (UDS-Tunnel)

Iniciaremos las máquinas UDS-Tunnel y procederemos a su configuración.

La primera tarea será asignar una dirección IP al servidor para poder acceder al asistente de configuración vía navegador. Para ello ejecutaremos el comando:

`uds ip set` *dirección_IP/mascara gateway hostname*

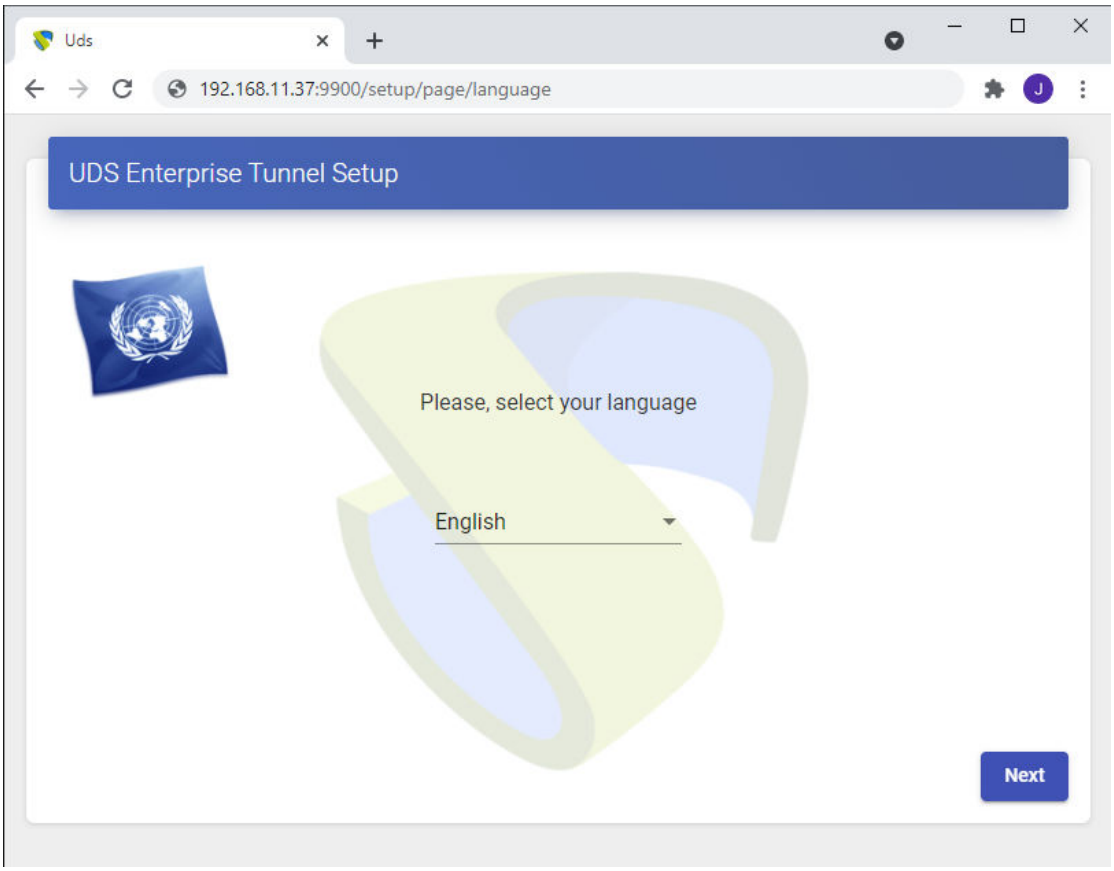


Después de indicar los datos IP, reiniciamos el servidor para aplicar los cambios.

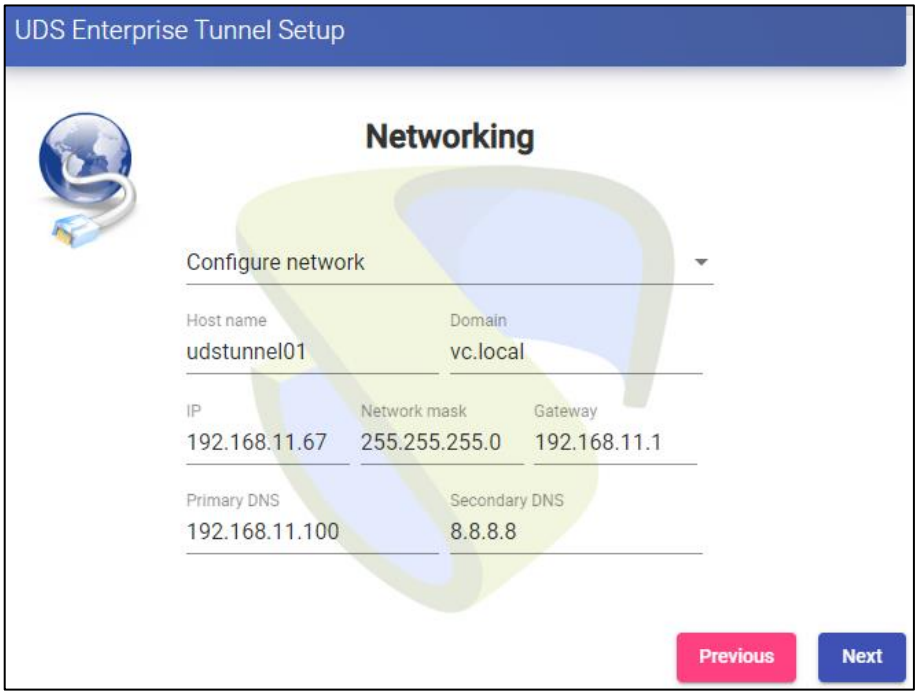
Si la red donde hemos desplegado el servidor Tunnel dispone de un servidor DHCP, este tomará una dirección IP vía DHCP que nos servirá para acceder al asistente de configuración.

A través de un navegador, accedemos a la URL indicada para iniciar el asistente de configuración del servidor Tunnel (en este ejemplo: http://192.168.11.37:9900).

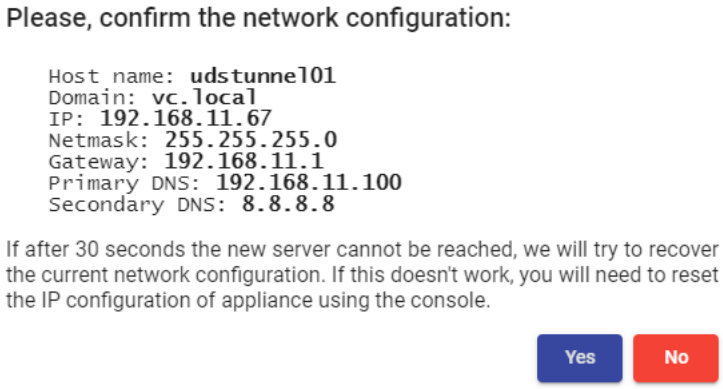
Seleccionamos el idioma del asistente de configuración:



En el apartado de redes, indicamos los datos IP, nombre y dominio (opcional) que tendrá nuestro servidor Tunnel:

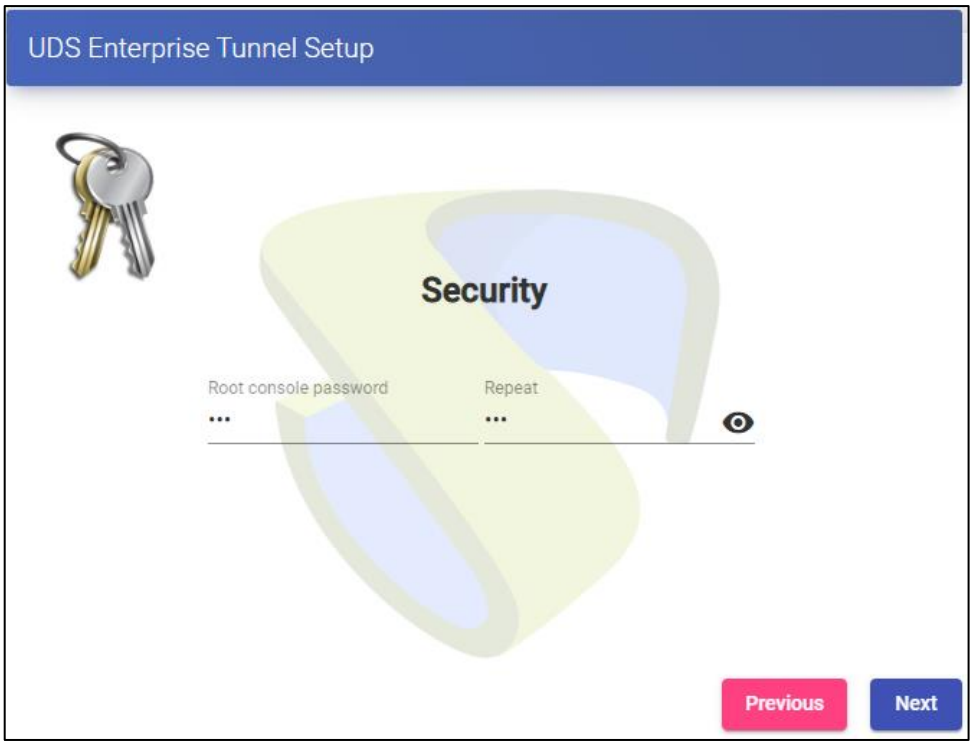


Confirmamos que los datos son correctos. Se procederá a aplicar los nuevos datos (en caso de acceder vía una dirección DHCP e indicar una dirección diferente, automáticamente se nos redirigirá, en el navegador, a la nueva dirección IP).



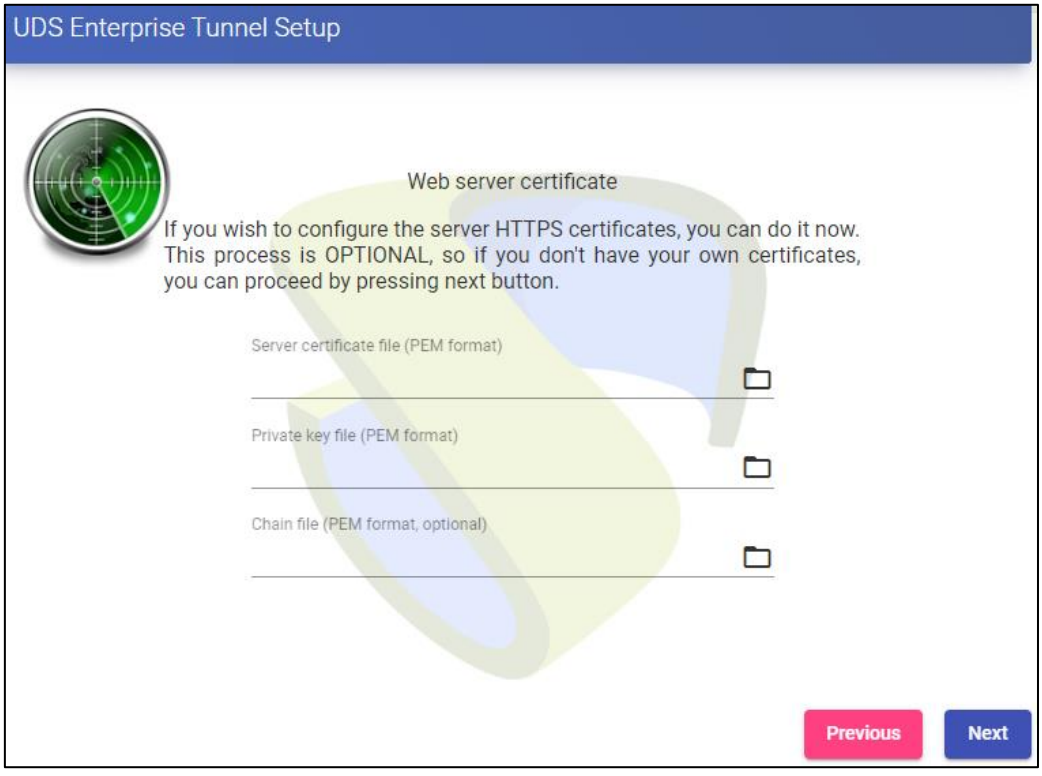
Seleccionamos el idioma del teclado, la zona horaria y opcionalmente podremos indicar un servidor NTP:

Indicaremos las credenciales del usuario root del S.O. Linux que aloja el servicio de UDS-Tunnel:

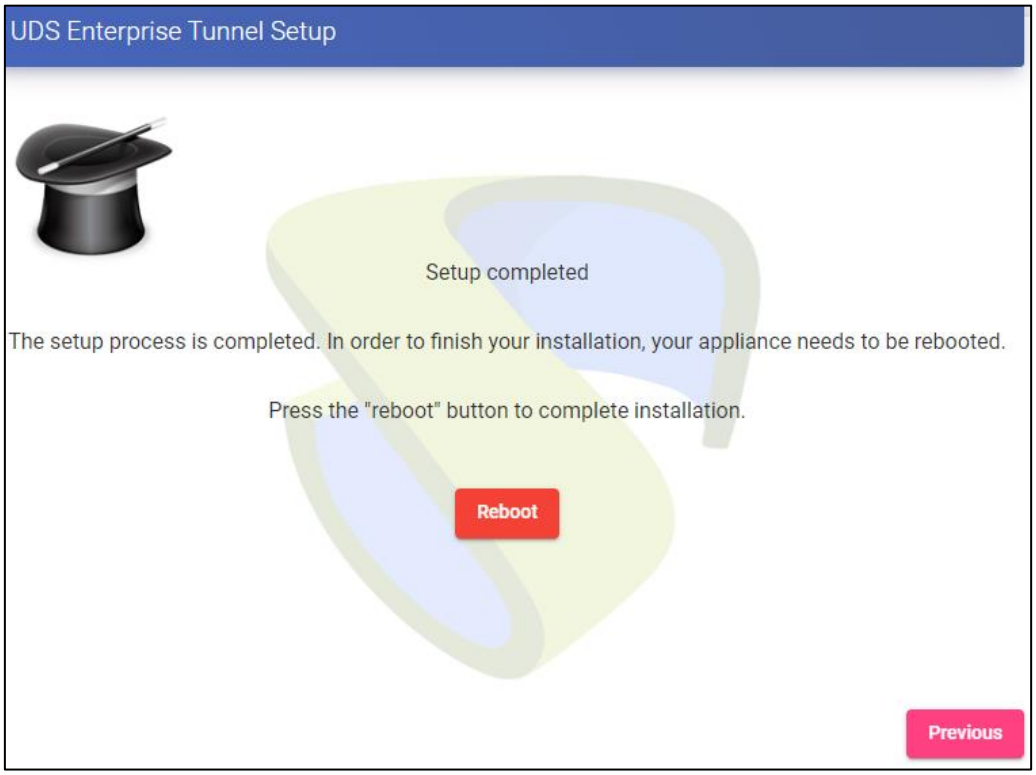


Seleccionamos cómo se realizará la conexión con el servidor UDS e indicamos su dirección IP. Como en este caso está configurado a través de un balanceador (HAProxy), dicha dirección será la IP virtual de balanceo configurada anteriormente en el servidor HAProxy usando el servicio Keepalived.

Podremos instalar los certificados en el servidor Tunnel para que las conexiones HTML5 dispongan de un certificado válido (en este ejemplo de dejaran los certificados autofirmados por defecto).



Reiniciaremos el servidor para finalizar su proceso de configuración.



Una vez reiniciado el servidor, ya estará disponible para ser utilizado en conexiones tunelizadas (RDP, X2Go, Spice, etc…) y HTML5.

Deberemos repetir todos los pasos anteriormente detallados en la segunda máquina UDS-Tunnel. Lógicamente los datos IP y nombre del segundo servidor serán diferentes, pero sí debemos conectar con la misma dirección IP virtual de balanceo para proporcionar acceso conexión con los servidores UDS.

Ambos servidores funcionarán en modo activo/activo, cada usuario que realice una conexión vía tunnel se conectarán de forma aleatoria a estos servidores. En caso de caída de uno de ellos, las conexiones de los usuarios que estén usando ese servidor se cortará, pero al volver a realizar dicha conexión accederá a través del servidor Tunnel activo de forma automática.

Sobre Virtual Cable

Virtual Cable desarrolla y comercializa UDS Enterprise mediante un modelo de suscripción, incluyendo soporte y actualizaciones, según el número de usuarios.

Además, Virtual Cable ofrece servicios profesionales para instalar y configurar UDS Enterprise.

Para más información, visite www.udsenterprise.com o envíenos un email a info@udsenterprise.