Centros de Datos y de Provisión de Servicios

Curso 2017-18

PRACTICA FINAL:

DESPLIEGUE DE UN SISTEMA CRM ESCALABLE

Última actualización: 29 de diciembre de 2017

Objetivo

- Creación de un escenario completo de despliegue de una aplicación fiable y escalable que integre los diversos contenidos impartidos en la asignatura.

Descripción

El objetivo de esta práctica es la implementación de la arquitectura completa de un sistema CRM escalable y fiable basado en el proyecto CRM utilizado en la asignatura IWEB y en la práctica 5. En el proyecto se utilizarán los elementos típicos de las arquitecturas actuales: firewall, balanceador de carga, servidores front-end corriendo la aplicación, bases de datos y servidores de almacenamiento, tal como aparece representado en la Figura 1.

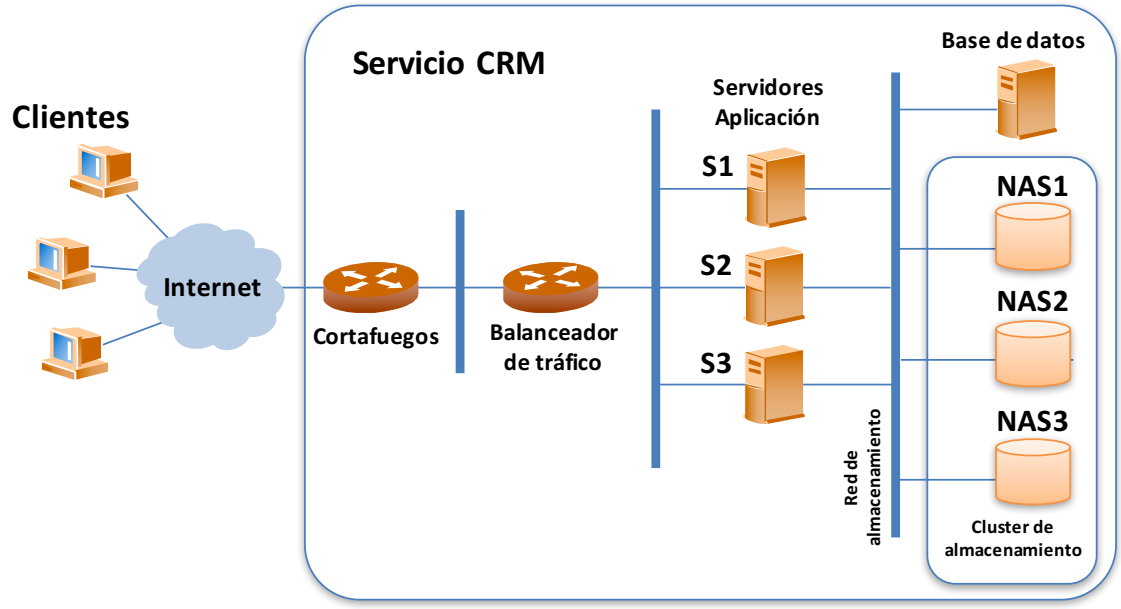

Figura 1: Arquitectura de la aplicación

La aplicación CRM se configurará para que utilice una base de datos PostgreSQL, que correrá en el servidor de bases de datos, y para que almacene las imágenes en el cluster de almacenamiento que se creará utilizando el sistema de ficheros distribuido Glusterfs. El balanceador de carga se ocupará de distribuir la carga entre los tres servidores que soportan la aplicación CRM (S1, S2 y S3) y el cortafuegos de entrada, basado en el software de Linux

FirewallBuilder, se ocupará de filtrar todo el tráfico proveniente de Internet y dejar pasar únicamente el destinado a la aplicación.

La arquitectura debe garantizar la escalabilidad de la aplicación, permitiendo ampliar fácilmente el número de servidores dedicados según crezca el número de usuarios. Por ello se parte de un sistema con un número determinado de servidores, pero se prevé añadir servidores (reales o virtuales) según crezca la demanda del servicio.

Los requisitos mínimos a cumplir son los siguientes:

- El servicio CRM debe estar accesible a los clientes en el puerto estándar de web (80).
- El balanceador debe balancear la carga entre todos los servidores utilizando un algoritmo round-robin.
- El cluster de almacenamiento, que se utilizará para almacenar las imágenes utilizadas por el CRM se debe configurar de forma que utilice GlusterFS y que replique la información entre los tres servidores (nas1, nas2 y nas3).
- La información manejada por el CRM se debe almacenar en el servidor de bases de datos, utilizando PostgreSQL (recomendado) u otro gestor de bases de datos soportado por el CRM. En cualquier caso, debe utilizarse una base de datos externa desplegada en el servidor destinado a tales efectos. Por lo tanto, no puede utilizarse SQLite.
- El firewall debe permitir únicamente el acceso mediante ping y al puerto 80 de TCP de la dirección balanceador de tráfico. Cualquier otro tráfico debe estar prohibido.

Adicionalmente, se proponen las siguientes partes opcionales:

1. Para facilitar la realización de la práctica, se proporciona un escenario virtual básico desarrollado con VNX. Alternativamente se puede realizar la practica utilizando Docker, Docker Compose y Flocker (https://github.com/clusterhq/flocker) para la gestión de los volúmenes.
2. Se puede instalar y configurar Nagios en la máquina virtual adicional 'nagios' para monitorizar toda la infraestructura. En este caso se configurará el firewall para que se pueda acceder a la monitorización desde los clientes.
3. Crear el procedimiento para añadir y configurar un nuevo servidor CRM adicional. Para se puede utilizar el escenario s4.xml, que añade un nuevo servidor s4 al escenario.
4. Utilizar un sistema de replicación de bases de datos para añadir resistencia a fallos.
5. Realizar mejoras en el algoritmo de balanceo de carga para seguir criterios de asignación en base a la carga, disponibilidad, etc. de los servidores.
6. Añadir un nuevo servidor de gestión (GES) al escenario que esté accesible por SSH desde los clientes, y cuyo objetivo sea poder gestionar los servidores del escenario remotamente. El acceso deberá realizarse por SSH y utilizando claves RSA (se deberá prohibir el acceso mediante nombre/clave de usuario).

Se pueden proponer otras partes opcionales aparte de las listadas aquí, aunque se recomienda consultarlas con los profesores previamente.

Entrega

Los alumnos deberán entregar un fichero zip en el que se incluya un documento donde se describa la arquitectura utilizada y la forma de instalar y configurar los servicios, así como los scripts y ficheros de configuración utilizados. En particular, el documento incluirá:

- Pasos seguidos en la implementación y despliegue de la solución, incluyendo una breve descripción de los paquetes y servicios que se ejecutarán en cada máquina.
- Una breve discusión sobre los puntos débiles de la arquitectura en cuanto a fiabilidad y escalabilidad, mencionando alguna solución a los problemas detectados.
- Partes opcionales implementadas.
- Breve descripción de los scripts de configuración utilizados o alternativamente alguna captura de pantalla con las configuraciones que hay que realizar para implementar la solución.
- Se indicará cómo cambiaría el despliegue en caso de desplegarlo en OpenStack y en Amazon AWS o Google Cloud.

En el examen oral de la práctica (cuyas fechas y turnos se publicarán en el moodle de la asignatura) se evaluará el correcto funcionamiento del servicio, la calidad de la solución adoptada en relación a los requisitos definidos en este documento y el conocimiento de los alumnos de las técnicas y herramientas usadas en el desarrollo de este trabajo.

Escenario virtual para realizar la práctica

Para facilitar la realización de la práctica se va a utilizar la herramienta Virtual Networks over linuX (VNX) de gestión de escenarios virtuales (http://vnx.dit.upm.es), que permite la automatización del arranque y configuración de escenarios virtuales similares a los creados en la práctica 3 y final de la primera parte, formados por máquinas virtuales interconectadas siguiendo una topología definida por el usuario. En particular, se ha preparado un escenario que implementa la arquitectura mostrada en la Figura 1, que de forma más precisa se representa en la Figura 2. El escenario está compuesto por dos máquinas cliente (C1 y C2), un cortafuegos (FW), un balanceador de carga (LB), tres servidores web (S1, S2 y S3) para alojar la aplicación CRM, tres servidores de disco (NAS1, NAS2 y NAS3) para crear el cluster de almacenamiento y un servidor de bases de datos, todos ellos implementados como máquinas virtuales ligeras (basadas en contenedores LXC).

La práctica podrá realizarse en el laboratorio o en un ordenador propio. En el caso del laboratorio, los escenarios pueden arrancarse directamente desde los puestos de prácticas, ya que dichos equipos tienen instalada la herramienta VNX.

En caso de realizar la práctica en ordenador propio, existen dos opciones:

- Utilizar la máquina virtual VirtualBox con Ubuntu 17.04 (http://idefix.dit.upm.es/download/cdps/CDPS2017-v1.ova) que tiene instaladas todas las herramientas necesarias para arrancar el escenario virtual.
- Utilizar un ordenador con Linux (Ubuntu 16.04 o más moderno) e instalar sobre él la herramienta VNX siguiendo las instrucciones que hay en http://web.dit.upm.es/vnxwiki/index.php/Vnx-install-ubuntu3. Esta opción es más costosa por la necesidad de instalar VNX, pero proporciona mejores prestaciones.

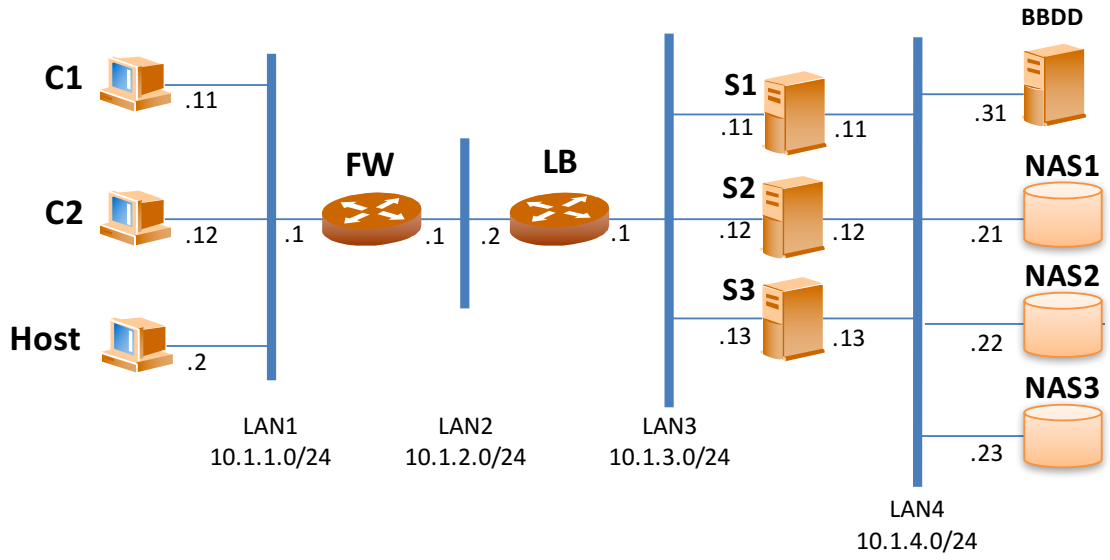

Figura 2: escenario virtual para realizar la práctica

Para arrancar el escenario virtual de la práctica siga las instrucciones siguientes:

1. **A) Si utiliza ordenador propio con VirtualBox:**

   - Descargue e instale VirtualBox desde http:///www.virtualbox.org. Debe instalar además el "VM VirtualBox Extension Pack" disponible también en la página de descargas. Si ya tiene instalado VirtualBox, se recomienda actualizarlo a la última versión.

   - Descargue la máquina virtual a su ordenador desde http://idefix.dit.upm.es/cdps/CDPS2017-v1.ova, impórtela a VirtualBox (Archivo->Importar servicio virtualizado) y arránquela.

   - Acceda a un terminal de la máquina virtual y descargue y descomprima el escenario:

```
wget http://idefix.dit.upm.es/cdps/pfinal/pfinal.tgz
sudo vnx --unpack pfinal.tgz
cd pfinal
bin/prepare-pfinal-vm
```

   B) Si utiliza ordenador propio con Linux y VNX, acceda a un terminal del PC y descargue el escenario y descomprímalo mediante:

```
wget http://idefix.dit.upm.es/cdps/pfinal/pfinal.tgz
sudo vnx --unpack pfinal.tgz
cd pfinal
bin/prepare-pfinal-vm
```

   C) Si utiliza el laboratorio, entre en su cuenta, acceda a un terminal, descargue el escenario y descomprímalo en el /mnt/tmp mediante:

```
cd /mnt/tmp
wget http://idefix.dit.upm.es/cdps/pfinal/pfinal.tgz
sudo vnx --unpack pfinal.tgz
cd pfinal
bin/prepare-pfinal-labo
```

   Nota importante: por restricciones de espacio en el laboratorio es necesario trabajar en el directorio /mnt/tmp, que se borra cada vez que se arranca el ordenador. Téngalo muy en cuenta para no perder información. Se recomienda copiar solo al /mnt/tmp el escenario virtual y guardar todos los scripts desarrollados en la cuenta del laboratorio.

2. Para gestionar el escenario virtual de la práctica, utilice los comandos siguientes:

   a. Arranque el escenario con:

```
sudo vnx -f pfinal.xml --create
```

   b. Para acceder a las máquinas virtuales del escenario puede hacerlo a través de las consolas que aparecen al arrancarlo o también mediante SSH (recomendado). Por ejemplo, para acceder a s1:

```
ssh root@s1
```

Nota: utilice la cuenta 'root' con clave 'xxxx' para acceder a las máquinas virtuales.

c. Para pararlo conservando los cambios realizados en las MVs:

```
sudo vnx –f pfinal.xml --shutdown
```

Tenga en cuenta lo siguiente: si realiza la práctica en la máquina virtual, los cambios se guardan en la misma, por lo que se conservarán en sucesivos arranques. En cambio, si realiza la práctica en el laboratorio, los cambios solo se conservarán mientras permanezca en el mismo puesto. Si cambia de puesto o utiliza el mismo puesto en días sucesivos, los cambios se perderán.

d. Para arrancar una máquina virtual parada con "--shutdown":

```
sudo vnx –f pfinal.xml --start
```

e. Para pararlo borrando los cambios realizados en las MVs:

```
sudo vnx –f pfinal.xml --destroy
```

f. Las operaciones de parada y arranque se pueden aplicar a de máquinas virtuales individuales (o listas de ellas separadas por comas) mediante la opción "-M". Por ejemplo, para parar y rearrancar el fw:

```
sudo vnx –f pfinal.xml --shutdown –M fw
sudo vnx –f pfinal.xml --start –M fw
```

g. Para mostrar el mapa del escenario:

```
sudo vnx –f pfinal.xml --show-map
```

3. Todas las máquinas virtuales del escenario utilizan el mismo sistema de ficheros (root-filesystem), que ya tiene instalados los principales paquetes necesarios. Si necesita instalar algún paquete adicional en alguna máquina virtual, puede instalarlo de la forma habitual (apt-get install …) ya que las máquinas virtuales tienen conexión a Internet. De todas formas, tenga en cuenta que los cambios realizados en las máquinas virtuales se perderán al parar el escenario con "--destroy".

   Es posible instalar nuevos paquetes de forma permanente mediante el procedimiento siguiente:

   - Parar el escenario de la práctica con la opción --destroy (IMPORTANTE: se perderán todos los cambios realizados en las MVs)

     ```
     sudo vnx –f pfinal.xml --destroy
     ```

   - Arranque una máquina virtual que utilice en modo directo y con conectividad a Internet la imagen de las MVs:

     ```
     sudo vnx --modify-rootfs filesystems/rootfs_lxc64-cdps --arch=x86_64
     ```

   - Una vez arrancada, active la red con:

     ```
     sudo dhclient eth0
     ```

- A continuación, instale todo lo que considere necesario. Una vez terminada la instalación, pare la máquina virtual con "halt -p". Todos los cambios que haya realizado quedarán permanentes en la imagen (guardada en el directorio `filesystems/rootfs_lxc64-cdps`) y, por tanto, serán visibles desde todas las máquinas virtuales.

4. Se recomienda que todas las configuraciones a realizar en las máquinas virtuales se realicen mediante uno o varios scripts externo (Python o shellscript) en el que se incluyan comandos "lxc-attach" o "cp" siguiendo las recomendaciones incluidas más abajo. De esta forma siempre será posible rearrancar en cualquier momento el escenario virtual y configurarlo desde el estado inicial.

5. Para mejorar el funcionamiento de la máquina virtual VirtualBox, se recomienda (siempre en función de los recursos que posea la máquina en la que se ejecute) aumentarle la memoria (más de 2Gb si es posible).

6. Es posible modificar el escenario de la práctica añadiendo, por ejemplo, más servidores si lo cree conveniente. Para ello, edite el fichero de definición del escenario (pfinal.xml) y replique la definición de las máquinas virtuales (bloques definidos por la etiqueta <vm>).

Ejecución de comandos en las máquinas virtuales

Es posible ejecutar comandos en las máquinas virtuales desde el host mediante el comando *lxc-attach*. Por ejemplo, el comando:

```
sudo lxc-attach --clear-env -n s1 -- service apache2 restart
```

accede a la máquina virtual s1 y ejecuta el comando "service apache2 restart".

Cuando se requiere ejecutar comandos que utilicen redirecciones es conveniente ejecutarlos de la siguiente forma:

```
sudo lxc-attach --clear-env -n s1 -- bash -c "echo 'texto' > fichero"
```

Para ejecutar múltiples comandos de una vez, pueden separarse mediante ";":

```
sudo lxc-attach --clear-env -n s1 -- bash -c "comando1; comando2; comando3"
```

Esta forma es muy útil cuando se requiere ejecutar comandos desde un determinado directorio (ej: "cd directorio; comando1; comando2"), ya que cada llamada a *lxc-attach* es independiente y si se cambia de directorio en una llamada, ese cambio no aplica a la siguiente.

Si el comando anterior se incluye en un shellscript en fichero, se puede utilizar la sintaxis siguiente para facilitar la lectura:

```
#!/bin/bash

sudo lxc-attach --clear-env -n s1 -- bash -c "
comando1;
comando2;
comando3
"
```

Si el comando a ejecutar incluye comillas, es necesario "escaparlas" precediéndolas con un "\" para que la Shell no las interprete y se pasen a la máquina virtual:

```
sudo lxc-attach --clear-env -n s1 -- bash -c "echo \"hola\" > /tmp/hola"
```

Si se prefiere utilizar un script en python en vez de un shellscript:

```
from subprocess import call

cmd_line = "sudo lxc-attach --clear-env -n s1 -- bash -c \"apt update; apt -y
install paquete\""
call (cmd_line, shell=True)

cmd_line = "sudo lxc-attach --clear-env -n s1 -- bash -c \"echo 'hola' >
fichero\""
call (cmd_line, shell=True)
```

Asimismo, es posible copiar ficheros directamente desde el host a una máquina virtual, dado que los sistemas de ficheros de las MVs ligeras se ven como directorios en el host. Por ejemplo, para copiar el fichero *ejemplo* desde el host al directorio /etc de la máquina virtual s1:

```
sudo cp ejemplo /var/lib/lxc/s1/rootfs/etc
```

Para realizar esa copia desde el laboratorio es necesario utilizar el comando "/mnt/vnx/repo/bin/cp2lxc" en vez de "cp":

```
sudo /mnt/vnx/repo/bin/cp2lxc ejemplo /var/lib/lxc/s1/rootfs/etc
```

Configuración del Cortafuegos

Para configurar el cortafuegos se recomienda utilizar la aplicación *fwbuilder*, que permite definir las reglas de una forma gráfica y después compilarlas a un script que se puede instalar para que se arranque cuando se arranca la máquina FW.

Para acceder a la consola de fwbuilder utilice el procedimiento siguiente desde un terminal:

```
ssh -X root@fw
fwbuilder fw.fwb &
```

El fichero fw.fwb contiene una configuración sencilla de ejemplo con una regla que permite el acceso web a s1. Para aprender a utilizar fwbuilder puede consultar el siguiente tutorial: http://www.fwbuilder.org/4.0/videos.shtml, o cualquier otro de los muchos disponibles en Internet.

Tenga en cuenta además que cuando se compilan las reglas definidas gráficamente, se crea un fichero *fw.fw* que es un shellscript que ejecuta todos los comandos necesarios para instalar las reglas. Ese fichero puede utilizarse para instalar las reglas sin necesidad de hacerlo a través del interfaz de fwbuilder.

Asimismo, se recomienda utilizar la herramienta *nmap* desde alguno de los clientes (C1, C2 o el propio host) para comprobar que efectivamente ñas reglas están correctamente configuradas.

Finalmente, también pueden utilizarse otras opciones más sencillas como el cortafuegos simplificado *ufw* disponible como paquete en Ubuntu.

Configuración de la base de datos

Para instalar y configurar la base de datos PostgreSQL que utilizarán los CRMs en el servidor BBDD, se deben ejecutar los siguientes pasos:

```
apt update
apt -y install postgresql

echo "listen_addresses='10.1.4.31'" >> /etc/postgresql/9.6/main/postgresql.conf
echo "host all all 10.1.4.0/24 trust" >> /etc/postgresql/9.6/main/pg_hba.conf

echo "CREATE USER crm with PASSWORD 'xxxx';" | sudo -u postgres psql
echo "CREATE DATABASE crm;" | sudo -u postgres psql
echo "GRANT ALL PRIVILEGES ON DATABASE crm to crm;" | sudo -u postgres psql
systemctl restart postgresql
```

Además, se deberá configurar adecuadamente la variable de entorno DATABASE_URL en los CRM para que estos accedan a la base de datos.

Configuración de GlusterFS

Se incluyen a continuación una serie de recomendaciones sobre la configuración del sistema de ficheros glusterfs en los servidores nasX y sobre como montar ese sistema de ficheros desde los servidores (sX).

1. Configuración de servidores de disco (nas)

   - Para simplificar, los servidores de disco (nas1-3) no tienen múltiples discos y, por tanto, no es necesario configurar ningún tipo de RAID en ellos. Se utilizará simplemente un directorio del sistema de ficheros de la máquina virtual (por ejemplo, /nas) que se exportará y sincronizará con el resto de servidores.

   - Para crear el cluster de servidores de disco replicados se pueden usar los comandos vistos en la práctica 3 y realizarlo enteramente desde el nas1. Esto es, añadimos los servidores al cluster mediante:

     ```
     gluster peer probe 10.1.4.XX
     ```

     Vemos el estado de los servidores con:

     ```
     gluster peer status
     ```

     Creamos un volumen con tres servidores que replican la información con:

     ```
     gluster volume create nas … (ver resto de parámetros en manual; es
     necesario usar la opción ‘force’)
     ```

     Arrancamos el volumen con:

     ```
     gluster volume start nas
     ```

     Vemos el estado de los volúmenes creados con:

     ```
     gluster volume info
     ```

Para agilizar la recuperación del volumen ante caídas de uno de los servidores, cambie el valor del timeout en todos los servidores (nas1-3) mediante:

```
gluster volume set nas network.ping-timeout 5
```

2. Configuración del montaje desde los servidores web (s1-s3)

   - Para acceder al sistema de ficheros exportado por los nasX desde los servidores de web se puede utilizar el comando mount. Por ejemplo:

     ```
     mkdir /mnt/nas
     mount -t glusterfs 10.1.4.21:/nas /mnt/nas
     ```

   - Una vez ejecutado ese comando, desde el directorio /mnt/nas del servidor web se verá el contenido del directorio exportado por los nas. Si se copia un fichero a ese directorio, se puede comprobar desde la consola de cada nas que fichero copiado se replica a todos los servidores.

   - **Importante:** la sincronización de ficheros entre los distintos nas solo funciona si los ficheros se copian/modifican/borran desde los servidores que montan el directorio compartido (s1-s3). Si modificáis los ficheros directamente desde los servidores nasX no funciona.

3. Se sugiere comprobar el funcionamiento del volumen gluster y su montaje desde los servidores web de la forma siguiente:

   - Acceda a las consolas de nas1, nas2 y nas3 y ejecute el comando siguiente:

     ```
     watch tree /nas
     ```

     que le permitirá ver en tiempo real el contenido del directorio compartido (/nas) de cada servidor.

   - Acceda a los servidores web (s1-s3) y copie y borre ficheros en el directorio /mnt/nas. Debería ver como esos ficheros aparecen y desaparecen de forma sincronizada en todos los nas.

   - Para ver cómo se comporta el sistema en caso de fallo de un servidor, desconecte un servidor cualquiera mediante:

     ```
     ifconfig eth1 down
     ```

   - Debería ver como las modificaciones de los ficheros en el directorio compartido dejan de sincronizarse con ese servidor. Si posteriormente vuelve a conectarlo, deberá ver como todos los cambios realizados mientras el servidor estaba caído se realizan y el servidor vuelve a estar sincronizado.

Instalación y configuración de la aplicación CRM

La instalación y configuración de la aplicación CRM en los servidores s1-s3 se realizará siguiendo las instrucciones disponibles en https://github.com/CORE-UPM/CRM_2017 y ya

utilizadas en la práctica 5. Únicamente se deberán tener en cuenta las siguientes consideraciones adicionales:

- Deberá instalarse nodejs (versión 8.9 o superior) y npm en s1-s3.
- Los comandos para aplicar las migraciones y ejecutar el seeder, solo deben ejecutarse desde uno de los servidores.
- Se configurará el CRM para que las imágenes subidas (por ejemplo, cuando se crea un nuevo usuario y se le asocia una foto) se copien a un directorio local de los servidores. Para ello simplemente no se configurará la variable de entorno CLOUDINARY_URL, lo que provoca que las imágenes se copien al directorio "public/upload" dentro del directorio del CRM.
- Se deberá hacer, además, que el directorio al que se copien las imágenes sea el directorio del cluster de almacenamiento que se monta mediante glusterfs, de forma que todos los servidores guarden las imágenes en el mismo sitio.
- Para arrancar el servidor CRM en background se recomienda utilizar la utilidad "forever". Para usarla, ejecute los comandos siguientes con *lxc-attach*:

```
cd /root/CRM_2017
npm install
npm install forever
export DATABASE_URL= …rellenar valores necesarios
npm run-script migrate_local      # solo en uno de los servidores
npm run-script seed_local         # solo en uno de los servidores
./node_modules/forever/bin/forever start ./bin/www
```

Configuración del balanceador de tráfico

Se recomienda utilizar el balanceador Crossroads ya utilizado en otras prácticas. Alternativamente pueden usarse otras opciones como HAproxy (http://www.haproxy.org/).

Instalación Opcional usando dockers y dockers compose

Para la ejecución de este escenario podemos usar:

- Un ordenador personal utilizando una instación de Docker (Windows, OSX, Linux).
- Una imagen que contenga Docker y Docker-Compose instalado.

Vamos a replicar el escenario usando contenedores dockers. Pera ello definiremos una serie de contenedores que son los que vamos a utilizar para la creación de nuestro escenario. Dichos contenedores serían:

- Cortafuegos (firewall)
- Balanceador de tráfico (loadbalancer)
- Servidores de Aplicaciones (webapp)

- Base de datos (postgress)
- Servidores de almacenamiento (storage). Nota: se proporciona una receta detallada de cómo arrancar dichos servidores. (https://www.1and1.com/cloud-community/learn/containers/docker/using-gluster-for-a-distributed-docker-storage-volume/)

Para ello definiremos una serie directorios con los nombres de cada una de las imágenes de ficheros “Dockerfile” con la creación de las imágenes de cada uno de los contenedores.

Se creará un fichero de configuración para Docker-Compose que arranque el escenario completo (una vez construidas cada una de las imágenes de los contenedores necesarios anteriormente descritos).

Para ello crearemos cuatros subredes y procederemos a su interconexión, tal como se muestra en la Figura 1.