UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática

Trabajo de fin de grado

Diseño e instalación de un servicio web de tienda virtual en alta disponibilidad

Autor:
Eloy García Martínez

Tutor:
Carmelo Rubén García Rodríguez
Catedrático de Escuela Universitaria
Ciencias de la Computación e Inteligencia Artificial

Las Palmas de Gran Canaria
Julio 2015

"A todas las personas que,
en mayor o menor medida,
han contribuido a que hoy
sea quien soy"

Agradecimientos

Quiero aprovechar este documento para agradecer a todas aquellas personas que a lo largo de la realización de estos estudios me han ayudado y apoyado.

Agradezco a cada uno de mis profesores todo lo que me han enseñado a lo largo de esta carrera. En especial a mi tutor, por darme la oportunidad de poder realizar este trabajo con él y por implicarse de la manera en la que lo ha hecho.

Del mismo modo quiero mostrar mi agradecimiento a toda mi familia por todo el apoyo dado durante estos años. Gracias a mis padres por todo el sacrificio que han realizado, sin ellos esto no hubiese sido posible.

Por último gracias a la persona más especial en mi vida, mi mujer, la que me acompaña día a día y me aguanta con su infinita paciencia.

A todos ellos, muchísimas gracias.

Resumen

El presente proyecto consiste en el diseño e instalación de un servicio web de tienda virtual ejecutándose en un entorno de computación en clúster con el objetivo de proporcionar alta disponibilidad y balanceo de carga del mencionado servicio. Toda la infraestructura requerida para este despliegue será virtual, utilizándose como plataforma de virtualización KVM.

Las tareas realizadas en este proyecto se organizan en dos bloques:

- Tareas para la creación de un clúster de balanceo de carga donde una máquina será la encargada de recibir las peticiones de los clientes y de redirigirlas a los servidores web según la carga de éstos.

- Tareas para la creación de un clúster de alta disponibilidad de manera que aseguramos que en caso de fallo siempre habrá un servidor de base de datos que pueda atender a las peticiones de los servidores web.

Abstract

This project involves the design and installation of a virtual shop web service running on cluster computing environment with the aim of providing high availability and load balancing of said service. All necessary infrastructure for this deployment will be virtual, using as virtualization platform KVM.

The tasks performed in this project are organized in two blocks:

- Tasks for creating of a load balancing cluster where a machine will be responsible for receiving requests from clients and redirect them to web servers according the load of these.

- Tasks for creating of a high-availability cluster so that we ensure that in case of failed there will always be a database server that can respond to requests from web servers.

Índice

1. Estado actual y objetivos. ... 13

1.1. Estado actual. ... 13

1.1.1. Modelos de virtualización. ... 13

1.1.2. Modelos de computación paralelos o distribuidos. ... 16

1.1.3. Tipos de clúster. ... 17

1.1.4. Sistemas de almacenamiento distribuidos. ... 18

1.2. Objetivos. ... 20

1.2.1. Objetivos generales del trabajo. ... 20

1.2.2. Objetivos personales. ... 20

2. Competencias. ... 21

2.1. CII01 ... 21

2.2. CII02 ... 21

2.3. CII04 ... 21

2.4. CII18 ... 21

2.5. TFG01. ... 22

3. Aportaciones. ... 23

4. Normativa y legislación. ... 24

4.1. Ámbito internacional ... 24

4.1.1. OCDE ... 24

4.1.2. Naciones Unidas. ... 24

4.2. Ámbito Europeo. ... 24

4.3. Ámbito nacional. ... 26

4.4. Normativas ISO. ... 26

4.5. Licencias Software Libre. ... 27

5. Metodología y plan de trabajo. ... 28

5.1. Metodología. ... 28

5.2. Plan de trabajo. ... 28

6. Análisis. ... 29

6.1. Obtención de requisitos. ... 29

6.2. Actores. ... 29

6.3. Casos de uso. ... 30

6.4. Análisis de las alternativas de balanceo de carga. ... 32

6.5. Análisis de los modos de balanceo de carga. ... 33

6.6. Análisis de alternativas de tienda virtual. ... 36

7. Diseño. 38
7.1. Diseño de la topología del clúster. 38
7.2. Identificación de los subsistemas de diseño. 39
7.3. Diseño de la infraestructura de red. 40
7.4. Diseño de la arquitectura de servicios. 41
7.4.1. Clúster de servidores web con balanceo de carga. 41
7.4.2. Clúster de servidores de base de datos en alta disponibilidad. 42
8. Requisitos hardware y software. 44
8.1. Requisitos hardware. 44
8.2. Requisitos software. 44
8.2.1. Sistema anfitrión. 44
8.2.2. Ordenador del laboratorio. 45
8.2.3. Máquinas virtuales. 45
9. Construcción. 47
9.1. Preparación del sistema anfitrión y del puesto de trabajo. 47
9.2. Creación de los recursos hardware. 48
9.2.1. Creación de las máquinas virtuales. 48
9.2.1.1. Creación de la primera máquina virtual. 48
9.2.1.2. Instalación del sistema operativo. 49
9.2.1.3. Clonación de las restantes máquinas virtuales. 50
9.2.2. Creación de una red virtual aislada. 52
9.2.3. Creación y configuración de las interfaces de red. 52
9.2.4. Creación de un volumen en la cabina de discos. 53
9.3. Creación y configuración del clúster de servidores web con balanceo de carga. 54
9.3.1. Creación y configuración de los balanceadores. 54
9.3.2. Creación y configuración de los servidores web. 59
9.4. Creación y configuración del clúster de base de datos en alta disponibilidad. 61
9.4.1. Instalación y configuración de los módulos software en los nodos del clúster. . 61
9.4.2. Configuración del cortafuegos. 63
9.4.3. Creación del clúster usando Conga. 64
9.4.4. Creación de un volumen lógico compartido. 66
9.4.4.1. Creación de un volumen físico. 66
9.4.4.2. Creación de un grupo de volúmenes. 67
9.4.4.3. Creación de un volumen lógico. 67
9.4.4.4. Creación de un sistema de archivos compartido. 68

9.4.5. Montaje del sistema de archivos distribuido. 68
9.4.6. Configuración del servicio Mysql en alta disponibilidad. 68
9.4.6.1. Creación de los recursos controlados por el clúster. 69
9.4.6.2. Creación de los dominios “failover”. 70
9.4.6.3. Creación de los grupos de servicios. 71
9.5. Instalación de la tienda virtual Prestashop. 72
9.5.1. Instalación de los módulos software necesarios en los servidores web. 72
9.5.2. Creación de la base de datos y su usuario. 72
9.5.3. Instalación. 73
10. Pruebas. 79
10.1. Pruebas de unidad. 79
10.1.1. Pruebas realizadas al clúster de servidores web con balanceo de carga. 79
10.1.1.1. Comprobación de la infraestructura de red. 79
10.1.1.2. Comprobación del correcto funcionamiento del balanceo de carga. 80
10.1.1.3. Comprobación de la recuperación ante fallos. 81
10.1.1.4. Comprobación del rendimiento. 81
10.1.2. Pruebas realizadas al clúster de servidores de base de datos en alta disponibilidad. 82
10.1.2.1. Comprobación de la infraestructura de red. 82
10.1.2.2. Comprobación del almacenamiento distribuido. 83
10.1.2.3. Comprobación de la recuperación ante fallos. 83
10.2. Pruebas de integración. 84
10.3. Pruebas del sistema. 84
11. Conclusiones y trabajos futuros. 85
11.1. Conclusiones. 85
11.2. Trabajo futuro. 85
12. Fuentes de información. 87
13. Anexos. 88
13.1. Pliego de condiciones técnicas. 88
13.2. Archivo de configuración del clúster de servidores web con balanceo de carga. 90
13.3. Archivo de configuración del clúster de servidor de base de datos en alta disponibilidad. 91
13.4. Script de generación de MACs. 91
13.5. Archivo de configuración XML de la máquina virtual base. 92

1. Estado actual y objetivos.

1.1. Estado actual.

Hoy en día la información es uno de los activos más importantes de las organizaciones y como tal hay que protegerlo. Por tanto asegurar su disponibilidad, integridad y confidencialidad se convertirá en una de las mayores prioridades para las empresas. Estas necesidades hacen que sea necesaria una mejor calidad de servicio, siendo de vital importancia que la información esté disponible el mayor tiempo posible. En un escenario así, sumado al hecho de que además se dispone de servidores cuyos recursos se encuentran infrautilizados, hace que sea preciso aplicar alguna técnica de virtualización. Surge así la necesidad de utilizar modelos de computación distribuida que proporcionen alta disponibilidad y alto rendimiento.

1.1.1. Modelos de virtualización.

De manera general se puede decir que virtualización es el efecto de abstraer los recursos de un computador con el objetivo de proporcionar acceso lógico a los recursos físicos. La virtualización separa de manera lógica la petición de algún servicio y los recursos físicos que realmente proporcionan el servicio.

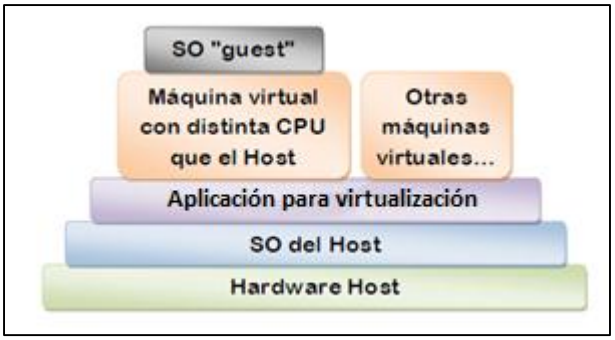
El núcleo de todo software de virtualización se conoce como hypervisor, también llamado monitor de máquina virtual (VMM). Éste es la parte principal de una máquina virtual que se encarga de manejar los recursos del sistema físico exportándolos a la máquina virtual.

Existen distintos modelos de virtualización dependiendo del recurso que se abstraiga (recurso individual o bien una plataforma) y de por quién sea usado ese recurso.

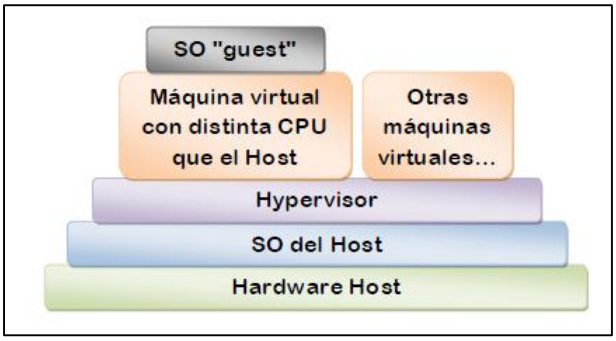
Virtualización de plataforma:

Consiste en la abstracción de todo el hardware subyacente de una plataforma de manera que múltiples instancias, copias del sistema operativo en formato ejecutable listas para funcionar, puedan ejecutarse de manera independiente, con la ilusión de que los recursos abstraídos les pertenecen en exclusiva. Existen los siguientes tipos y paradigmas de virtualización:

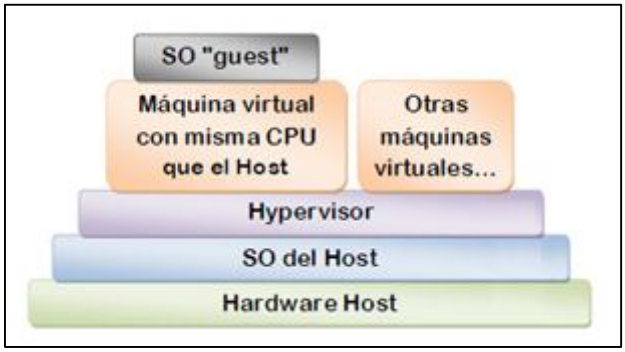
- **Sistemas operativos invitados.**
  Sobre una aplicación para virtualización que corre sobre la instancia de un sistema operativo se permite la ejecución de máquinas virtuales con sistemas operativos independientes. No hace uso de hipervisor u otra capa de virtualización.
  Si la aplicación de virtualización implementa traducción del juego de instrucciones o emulación podrán ser ejecutadas máquinas virtuales cuyo sistema operativo, utilidades y aplicaciones hayan sido compiladas para hardware y juego de instrucciones diferentes al de la máquina física anfitriona, en caso contrario no.



- **Emulación.**
  Un emulador que replica una arquitectura hardware al completo permite que se ejecuten sobre él máquinas virtuales. Por lo tanto se permite la ejecución de sistemas operativos y aplicaciones distintos al instalado físicamente en la máquina que ejecuta el emulador.

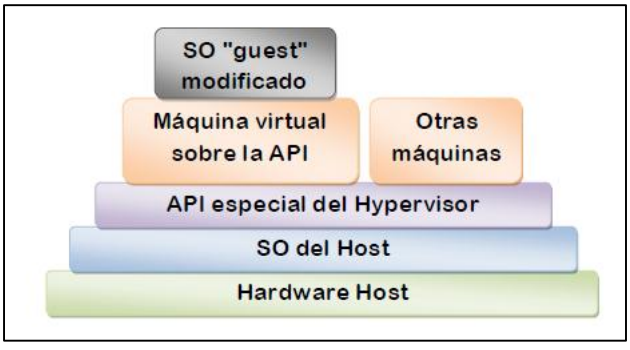


- **Virtualización completa o nativa.**
  Un hipervisor media entre los sistemas invitados y el anfitrión, el cual incluye código que emula el hardware subyacente, si es necesario, para las máquinas virtuales, por lo que es posible ejecutar cualquier sistema operativo sin modificar, siempre que soporte el hardware subyacente. El código de emulación puede provocar pérdida en el rendimiento.

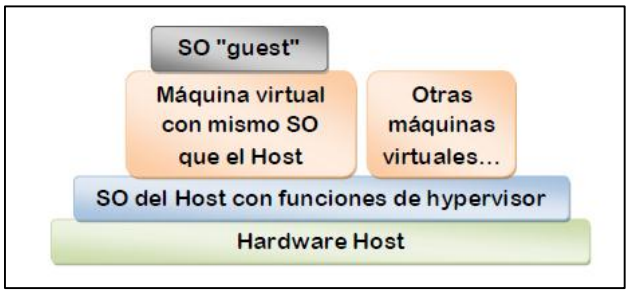


- **Paravirtualización.**

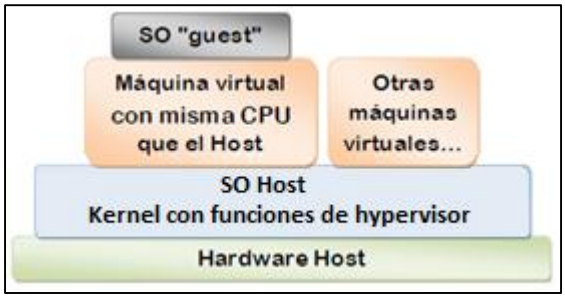
Similar a la virtualización completa porque introduce hipervisor como capa de virtualización, pero no incluye emulación del hardware. El hypervisor ofrece una API especial que sólo puede usarse mediante la modificación del SO invitado. Esto elimina la necesidad de captura de instrucciones privilegiadas o conflictivas por parte del hipervisor, mejorando el rendimiento hasta obtenerlo casi similar a un sistema no virtualizado. Las librerías y utilidades ejecutadas por las máquinas virtuales deben estar compiladas para el mismo hardware y juego de instrucciones que el de la máquina física anfitriona.



- **Virtualización a nivel del sistema operativo.**
  Las máquinas se virtualizan sobre el propio sistema operativo, sin necesidad de introducir una capa intermedia de virtualización. Por lo tanto, simplemente aísla las máquinas independientes, que comparten el mismo sistema operativo. Al compartir el mismo núcleo, éstas no pueden correr sistemas operativos diferentes y además las librerías y utilidades ejecutadas deben estar compiladas para el mismo hardware y juego de instrucciones que el de la máquina física.



- **Virtualización a nivel del kernel.**
  Convierte el núcleo Linux en hipervisor utilizando un módulo, el cual permite ejecutar máquinas virtuales y otras instancias de sistemas operativos en el espacio de usuario del núcleo Linux anfitrión. Las librerías, aplicaciones y sistemas operativos de las máquinas virtuales deben ser soportados por el hardware subyacente del anfitrión.



Virtualización de recursos.

En este caso el recurso abstraído es un recurso individual de un computador. Dentro de este tipo de virtualización tenemos los siguientes modelos.

- Encapsulación.
- Memoria virtual.
- Virtualización de almacenamiento.
- Virtualización de red.
- Unión de interfaces de red.
- Virtualización de Entrada/Salida.
- Virtualización de memoria.

Virtualización de aplicaciones.

Las aplicaciones son ejecutadas encapsuladas sobre el sistema operativo de manera que aunque creen que interactúan con él de la manera habitual, en realidad no lo hacen, sino que lo hacen bien con una máquina virtual de aplicación o con algún software de virtualización.

Virtualización de escritorio.

Consiste en la manipulación de forma remota del escritorio de usuario (aplicaciones, archivos, datos), que se encuentra separado de la máquina física, almacenado en un servidor central remoto en lugar de en el disco duro del computador local.

1.1.2. Modelos de computación paralelos o distribuidos.

Ahora que ya hemos visto qué tipos de virtualización hay, vamos a conocer los principales modelos de computación distribuidos que existen:

Cluster computing.

Este tipo de computación consiste en un grupo de computadores de relativo bajo coste conectados entre sí mediante un sistema de red de alta velocidad y un software que realiza la distribución de la carga de trabajo entre los equipos. Por lo general, este tipo de sistemas cuentan con un centro de almacenamiento de datos único.

En función del objetivo perseguido tenemos los siguientes tipos de clúster:

- **Alto rendimiento**: Son clústeres cuyo objetivo de diseño es la de ejecutar tareas que requieren de una gran capacidad computacional, grandes cantidades de memoria, o ambos a la vez.
- **Alta disponibilidad**: Son clústeres cuyo objetivo de diseño es el de proveer disponibilidad y confiabilidad. Estos clústeres tratan de brindar la máxima disponibilidad de los servicios que ofrecen. La confiabilidad se provee mediante software que detecta fallos y permite recuperarse frente a los mismos, mientras que en hardware se evita tener un único punto de fallos.
- **Alta eficiencia**: Son clústeres cuyo objetivo de diseño es el ejecutar la mayor cantidad de tareas en el menor tiempo posible. Existe independencia de datos entre las tareas individuales. El retardo entre los nodos del clúster no es considerado un gran problema.

Grid computing.

La computación en grid o en malla es un nuevo paradigma de computación distribuida en el que todos los recursos de un número indeterminado de computadoras son englobados para ser tratados como un único superordenador de manera transparente.

Estas computadoras englobadas no están conectadas o enlazadas firmemente, es decir no tienen por qué estar en el mismo lugar geográfico.

Cloud computing.

La computación en la nube es un término general para denominar cualquier cosa que tenga que ver con la provisión de servicios de hospedaje a través de Internet. Estos servicios se dividen en tres grandes categorías:

- Infraestructura como servicio (IaaS).
- Plataforma como servicio (PaaS).
- Software como servicio (SaaS).

Los servicios en la nube tienen tres características bien diferenciadas:

- La tarificación se realiza en función del uso.
- El servicio es elástico, ya que el usuario puede usar tanto como quiera y en el momento que lo desee.
- El servicio es gestionado en su totalidad por el proveedor.

1.1.3. Tipos de clúster.

En este apartado analizaremos las características de los distintos tipos de clúster que existen en la actualidad.

Clúster de alto rendimiento.

Un clúster de alto rendimiento es un conjunto de ordenadores que está diseñado para dar altas prestaciones en cuanto a capacidad de cálculo. Los motivos para utilizar un clúster de alto rendimiento son:

- El tamaño del problema por resolver
- El precio de la máquina necesaria para resolverlo.

Por medio de un clúster se pueden conseguir capacidades de cálculo superiores a las de un ordenador más caro que el costo conjunto de los ordenadores del clúster.

Clúster de balanceo de carga.

Un clúster de balanceo de carga o de cómputo adaptativo está compuesto por uno o más ordenadores (llamados nodos) que se encargan de capturar las peticiones de servicio que recibe el clúster, y los ordenadores cuya función es procesar las peticiones que los anteriores les reparten. Las características más destacadas de este tipo de clúster son:

- Se puede ampliar su capacidad fácilmente añadiendo más ordenadores al clúster.
- Robustez. Ante la caída de alguno de los ordenadores del clúster el servicio se puede ver mermado, pero mientras haya ordenadores en funcionamiento, éstos seguirán dando servicio.

Clúster de alta disponibilidad.

Un clúster de alta disponibilidad consiste en un conjunto de ordenadores conectados entre sí de tal manera que cuando se produce un fallo del hardware o de las aplicaciones de alguna de las máquinas del clúster, el software de alta disponibilidad es capaz de arrancar automáticamente los servicios que han fallado en cualquiera de las otras máquinas del clúster. Las principales características que tiene son:

- Se puede ampliar su capacidad fácilmente añadiendo más ordenadores al clúster.
- Robustez. Ante la caída de alguno de los ordenadores del clúster, el servicio migra a un ordenador de respaldo, por lo que se garantiza la integridad de la información y además evita molestias a los usuarios, que no tienen por qué notar que se ha producido un problema.

1.1.4. Sistemas de almacenamiento distribuidos.

Las necesidades de almacenamiento de datos se han ido incrementando continuamente tanto para usuarios como en las empresas. Cualquier información es susceptible de ser digitalizada y almacenada; por tanto se necesitan manejar mayores volúmenes de datos y al mismo tiempo se hace necesario mantener alguna copia adicional de la información más sensible.

Los sistemas de almacenamiento distribuido permiten centralizar la información, realizar una mejor gestión del espacio y crear sistemas redundantes y de alta disponibilidad. Para ello se establece una red dedicada de alto rendimiento para conectar directamente los dispositivos de almacenamiento, lo que permite a los archivos y datos ser directamente transferidos entre dispositivos de almacenamiento y maquinas cliente, saltándose el tradicional cuello de botella del servidor y el control de la red. De esta forma se consigue separar el tráfico normal de la red del tráfico relacionado con los medios de almacenamiento.

DAS (Direct Attached Storage).

Es el método tradicional de almacenamiento y el más sencillo. Consiste en conectar el dispositivo de almacenamiento directamente al servidor o estación de trabajo, es decir, físicamente conectado al dispositivo que hace uso de él.

NAS (Network Attached Storage).

Los SAN como su nombre lo indica son dispositivos dedicados de almacenamiento en red, que utilizan los recursos de la redes LAN para prestar el servicio de almacenamiento. Estos dispositivos integran un sistema operativo que ha sido específicamente desarrollado para cumplir con el propósito de alojar y compartir archivos dentro de una red LAN.

Las características avanzadas que proporcionan los NAS permiten establecer políticas de acceso y redundancia sobre los datos con el objeto de mantener la seguridad de los mismos.

SAN (Storage Area Network).

Las SAN son sistemas dedicados de almacenamiento en red que acceden a los recursos mediante conexiones directas entre el servidor y los dispositivos de almacenamiento de manera que no causan impacto sobre la LAN.

El alojamiento y compartición de archivos se realiza mediante un disco virtual que conforman todos los dispositivos que pertenecen a la red y es en esté, donde la información se consolida y se comparte entre los diferentes servidores.

Un aspecto importante de las SAN es que al estar los datos distribuidos entre los dispositivos de la red, estos tienen una alta disponibilidad y redundancia en caso de fallo en cualquier componente del sistema.

RAID (Redundant Array of Independent Disks).

Son sistemas de almacenamiento basados en discos duros que tienen la capacidad de actuar colectivamente para replicar los datos en distintas unidades, con el fin de garantizar la disponibilidad y la persistencia de la información almacenada en cada uno de los discos que lo conforman.

Nivel	Confiabilidad	Rendimiento	Disponibilidad
RAID 0	• No proporciona tolerancia a fallos.	• Mejora la tasa detransferencia y el tiempo de acceso a los datos.	• El sistema deja de funcionar si hay una unidad de disco en falla.
RAID 1	• Protege la información en caso de falla.	• Mejora la lectura de los datos.	• Evita interrupciones por fallas en las unidades.
RAID 2	• El uso del código Hamming permite detectar y corregir errores.	• Mejora la operación de aplicaciones con alta tasa de transferencia.	• Usa múltiples discos dedica dos que permiten redundancia de datos.
RAID 3	• El disco de paridad permite reconstruir la información.	• Elevada tasa de transferencias secuenciales.	• Si falla un disco el sistema puede seguir en funciona miento.
RAID 4	• Es ideal para almacenar ficheros de gran tamaño.	• Durante las operaciones de lectura-escritura las unidades de disco son accesadas de forma individual.	• Es tolerante a fallos ya que se puede recuperar los datos de un disco averiado en tiempo real.
RAID 5	• Distribuye los datos de paridad entre todas las unidades de disco.	• La velocidad de transferencia de datos es alta.	• Es tolerante a fallos con una unidad de disco averiada.
RAID 6	• Cada dato de paridad es redundante y distribuido en dos unidades de disco diferentes.	• Las operaciones de escritura esultan más lentas que las de lectura de datos.	• Es tolerante a fallos con dos unidades de discos averiadas.

1.2. Objetivos.

1.2.1. Objetivos generales del trabajo.

El objetivo principal de este trabajo es crear un entorno de alta disponibilidad y balanceo de carga para desplegar un servicio de tienda virtual utilizando para ello la plataforma de virtualización KVM. Las tareas que se van a llevar a cabo para cumplir dicho objetivo son las siguientes:

- Desarrollar el catálogo de requisitos del sistema a construir.
- Elaborar el diseño de la solución.
- Instalar y configurar los elementos virtuales de infraestructura: computo, almacenamiento y de red requeridos.
- Instalar y configurar los componentes software que permiten el cómputo en clúster con el objetivo de proporcionar servicios en alta disponibilidad y balanceo de carga.
- Diseño y creación de los distintos elementos que posibilitan la ejecución del servicio web de tienda virtual en alta disponibilidad.
- Diseño y creación de los distintos elementos que posibilitan la ejecución del servicio web de tienda virtual con balanceo de carga.
- Diseño y creación del servicio web de tienda virtual.

1.2.2. Objetivos personales.

El Trabajo de Fin de Grado pretende ser la muestra de los conocimientos y habilidades adquiridas a lo largo de toda la titulación. Asimismo, la elaboración de éste será una buena oportunidad para:

- Consolidar y aumentar los conocimientos sobre las tecnologías de la virtualización
- Consolidar y aumentar los conocimientos sobre las tecnologías para la alta disponibilidad.
- Consolidar y aumentar los conocimientos sobre las tecnologías para el balanceo de carga.

2. Competencias.

A continuación se describen las competencias que han sido cubiertas durante la realización de este proyecto, indicando en cada caso cómo han sido satisfechas.

2.1. CII01.

"Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y a la legislación y normativa vigente".

A lo largo de la elaboración de este documento, más concretamente en los capítulos 6, 7, 8 y 9, ha quedado demostrada la capacidad para analizar las necesidades del cliente con el objetivo de diseñar, desarrollar y evaluar una solución que cumpla con todas ellas.

2.2. CII02.

"Capacidad para planificar, concebir, desplegar y dirigir proyectos, servicios y sistemas informáticos en todos los ámbitos, liderando su puesta en marcha y su mejora continua y valorando su impacto económico y social".

A la hora de realizar un proyecto, la planificación debe afrontarse de manera adecuada para garantizar el cumplimiento de los plazos de entrega y que la calidad del producto sea el deseado.

Puesto que durante este proyecto se ha realizado una buena planificación, se ha podido concluir en la fecha planificada ajustándose a la definición de éste.

2.3. CII04.

"Capacidad para elaborar el pliego de condiciones técnicas de una instalación informática que cumpla los estándares y normativas vigentes".

En el anexo se adjunta el pliego de condiciones técnicas para la contratación de los servicios para la puesta en marcha de un servicio web de tienda virtual en alta disponibilidad.

2.4. CII18.

"Conocimiento de la normativa y la regulación de la informática en los ámbitos nacional, europeo e internacional".

En el capítulo 4 Normativa y legislación se indican las principales normativas y leyes que afectan a este proyecto tanto a nivel nacional como a nivel europeo e internacional. Principalmente son aquellas relacionadas con el comercio electrónico, protección de datos personales y firma electrónica.

2.5. TFG01.

"Ejercicio original a realizar individualmente y presentar y defender ante un tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas de la Ingeniería en Informática de naturaleza profesional en el que se sinteticen e integren las competencias adquiridas en las enseñanzas".

Esta competencia queda cubierta con la elaboración de este Trabajo de Fin de Grado y su posterior presentación ante el tribunal evaluador.

3. Aportaciones.

Este proyecto pretende crear un entorno de alta disponibilidad y balanceo de carga donde las organizaciones puedan desplegar sus tiendas virtuales. Es de vital importancia que estos servicios estén disponibles de forma continuada ya que cualquier tipo de interrupción afectaría a sus ventas y a lo que es más importante, a su imagen.

La solución elegida tiene las siguientes características:

Económica:

Al hacer uso de la virtualización, las empresas pueden elevar las tasas de utilización de sus servidores haciendo un uso más eficiente de sus recursos y del capital de la empresa. Por tanto se necesitarán menos máquinas físicas, lo que conlleva un menor consumo energético de forma directa e indirectamente ya que disminuye el gasto en refrigeración. Además también se ahorra en espacio ya que menos servidores implican un menor tamaño del Data Center.

A parte, durante todo el desarrollo se ha utilizado software libre, lo que ha abaratado considerablemente la solución elegida.

Escalable:

En caso de que el número de usuarios de la tienda virtual aumentara hasta hacer insuficientes los recursos actuales de alguno de los servicios (servidores web o servidores de base de datos), éstos se pueden ampliar de forma muy sencilla y rápida, basta con clonar la máquina virtual que contiene dicho servicio y añadírsela al clúster.

Fiable:

Existe suficiente infraestructura para garantizar que si falla alguna de las máquinas que ofrecen el servicio, éste siga estando disponible, haciendo que los usuarios no se percaten en ningún momento de lo ocurrido.

4. Normativa y legislación.

En este capítulo veremos las principales normativas y leyes vigentes que afectan a este proyecto. Se comenzará dando una visión en el ámbito internacional conociendo las leyes y recomendaciones de organismos como la OCDE y Naciones Unidas. Luego pasaremos al ámbito europeo para terminar en nuestro propio país.

4.1. Ámbito internacional.

Desde la perspectiva jurídica, apenas existe legislación internacional. Lo que sí existen son directrices y recomendaciones promovidas por la Organización para la Cooperación y el Desarrollo Económicos (OCDE) y Naciones Unidas (ONU).

4.1.1. OCDE.

- Directrices de la OCDE sobre protección de la privacidad y flujos transfronterizos de datos personales.

- Directrices de la OCDE para la Protección de los Consumidores de Prácticas Comerciales Transfronterizas Fraudulentas y Engañosas.

- Recomendación del consejo de la OCDE relativa a los lineamientos para la protección al consumidor en el contexto del comercio electrónico.

4.1.2. Naciones Unidas.

- Ley Modelo de la CNUDMI sobre Comercio Electrónico (1996).

- Directrices para la regulación de los archivos de datos personales informatizados.

4.2. Ámbito Europeo.

En Europa durante los últimos años se ha ido conformando un marco normativo que regula las obligaciones que deben cumplir tanto las administraciones públicas como los organismos privados en el uso y gestión de la información.

Las principales directivas y reglamentos que afectan a este proyecto son:

- Reglamento (UE) nº 910/2014 del Parlamento Europeo y del Consejo, de 23 de julio de 2014, relativo a la identificación electrónica y los servicios de confianza para las transacciones electrónicas en el mercado interior y por la que se deroga la Directiva 1999/93/CE.

- Reglamento (UE) Nº 611/2013 de la Comisión, de 24 de junio de 2013, relativo a las medidas aplicables a la notificación de casos de violación de datos personales en el marco de la Directiva 2002/58/CE del Parlamento Europeo y del Consejo sobre la privacidad y las comunicaciones electrónicas.

- Directiva 2009/136/CE del Parlamento Europeo y del Consejo, de 25 de noviembre de 2009, por la que se modifican la Directiva 2002/22/CE relativa al servicio universal y los derechos de los usuarios en relación con las redes y los servicios de comunicaciones electrónicas, la Directiva 2002/58/CE relativa al tratamiento de los datos personales y a la protección de la intimidad en el sector de las comunicaciones electrónicas y el Reglamento (CE) no 2006/2004 sobre la cooperación en materia de protección de los consumidores.

- Directiva 2006/24/CE, del Parlamento Europeo y del Consejo de 15 de marzo de 2006, sobre la conservación de datos generados o tratados en relación con la prestación de servicios de comunicaciones electrónicas de acceso público o de redes públicas de comunicaciones y por la que se modifica la Directiva 2002/58/CE.

- Directiva 2002/58/CE del Parlamento Europeo y del Consejo de 12 de julio de 2002, relativa al tratamiento de los datos personales y a la protección de la intimidad en el sector de las comunicaciones electrónicas (Directiva sobre privacidad y las comunicaciones electrónicas).

- Directiva 2002/22/CE del Parlamento Europeo y del Consejo, de 7 de marzo de 2002, relativa al servicio universal y los derechos de los usuarios en relación con las redes y los servicios de comunicaciones electrónicas (Directiva servicio universal).

- Directiva 2002/20/CE, del Parlamento Europeo y del Consejo, de 7 de marzo de 2002, relativa a la autorización de redes y servicios de comunicaciones electrónicas (Directiva de autorización).

- Directiva 2002/19/CE, del Parlamento Europeo y del Consejo, de 7 de marzo de 2002, relativa al acceso a las redes de comunicaciones electrónicas y recursos asociados, y a su interconexión (Directiva de acceso).

- Directiva 2000/31/CE, del Parlamento Europeo y del Consejo, de 8 de junio de 2000, relativa a determinados aspectos jurídicos de los servicios de la sociedad de la información, en particular el comercio electrónico en el mercado interior (Directiva sobre el comercio electrónico).

- Directiva 95/46/CE del Parlamento Europeo y del Consejo de 24 de octubre de 1995 relativa a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos.

4.3. Ámbito nacional.

Los primeros rastros legales que nos encontramos en España relacionados con la seguridad de la información se localizan en la Constitución de 1978, donde nos podemos encontrar el siguiente artículo:

Artículo 18 Derecho a la intimidad. Inviolabilidad del domicilio.

1. Se garantiza el derecho al honor, a la intimidad personal y familiar y a la propia imagen.

2. El domicilio es inviolable. Ninguna entrada o registro podrá hacerse en él sin consentimiento del titular o resolución judicial, salvo en caso de flagrante delito.

3. Se garantiza el secreto de las comunicaciones y, en especial, de las postales, telegráficas y telefónicas, salvo resolución judicial.

4. La ley limitará el uso de la informática para garantizar el honor y la intimidad personal y familiar de los ciudadanos y el pleno ejercicio de sus derechos.

A continuación, indicaremos las principales leyes que afectan a este trabajo:

- Ley 59/2003, de 19 de diciembre, de firma electrónica.
- Ley 34/2002, de 11 de julio, de servicios de la sociedad de la información y de comercio electrónico.
- Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.

4.4. Normativas ISO.

La serie de normas ISO/IEC 27000 son estándares para el ámbito de seguridad de la información publicados por la Organización Internacional para la Estandarización (ISO) y la Comisión Electrotécnica Internacional (IEC).

La serie consiste en un conjunto de recomendaciones y buenas prácticas que se deben seguir para desarrollar, implementar y mantener un Sistema de Gestión de la Seguridad de la Información (SGSI).

ISO/IEC 27000: Es un vocabulario estándar para el SGSI.

ISO/IEC 27001: Es la certificación que deben obtener las organizaciones.

ISO/IEC 27002: Es código de buenas prácticas para la gestión de seguridad de la información.

ISO/IEC 27003: Son directrices para la implementación de un SGSI.

ISO/IEC 27004: Son métricas para la gestión de seguridad de la información.

ISO/IEC 27005: Trata la gestión de riesgos en seguridad de la información.

ISO/IEC 27006: Requisitos para la acreditación de las organizaciones que proporcionan la certificación de los sistemas de gestión de la seguridad de la información.

ISO/IEC 27007: Es una guía para auditar al SGSI.

4.5. Licencias Software Libre.

Software libre es el software que respeta la libertad de los usuarios y la comunidad. A grandes rasgos, significa que los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software.

Un programa es software libre si los usuarios tienen las cuatro libertades esenciales:

- La libertad de ejecutar el programa como se desea, con cualquier propósito.
- La libertad de estudiar cómo funciona el programa, y cambiarlo para que haga lo que usted quiera. El acceso al código fuente es una condición necesaria para ello.
- La libertad de redistribuir copias para ayudar a su prójimo.
- La libertad de distribuir copias de sus versiones modificadas a terceros. Esto le permite ofrecer a toda la comunidad la oportunidad de beneficiarse de las modificaciones. El acceso al código fuente es una condición necesaria para ello.

Podemos agrupar la mayoría de las licencias de software libre en tres grandes grupos: licencias copyleft fuertes, copyleft débiles y permisivas.

- Copyleft fuertes: Estás licencias garantizan que el software original y todas sus modificaciones siempre sean software libre.

- Copyleft débil: Son aquellas licencias que si bien mantienen su propio código fuente bajo copyleft, permiten el uso, integración y redistribución en programas de ordenador bajo otras licencias.

- Licencias permisivas: Son aquellas licencias que no imponen ninguna condición particular respecto a la redistribución del software, aunque incluyen obligaciones de mantener los avisos legales y las limitaciones de garantía y exoneración de responsabilidad.

5. Metodología y plan de trabajo.

5.1. Metodología.

Para la realización de este proyecto se ha tenido en cuenta tanto el manual operativo de trabajos de fin de título de la EII como Métrica v.3, que es una metodología de planificación, desarrollo y mantenimiento de sistemas de información, promovida por el Ministerio de Hacienda y Administraciones Públicas del Gobierno de España que tiene la siguiente estructura:

- Introducción.
- Planificación de Sistemas de Información (Proceso PSI).
- Estudio de Viabilidad del Sistema (Proceso EVS).
- Análisis del Sistema de Información (Proceso ASI).
- Diseño del Sistema de Información (Proceso DSI).
- Construcción del Sistema de Información (Proceso CSI).
- Implantación y Aceptación del Sistema (Proceso IAS).
- Mantenimiento del Sistema de Información (Proceso MSI).

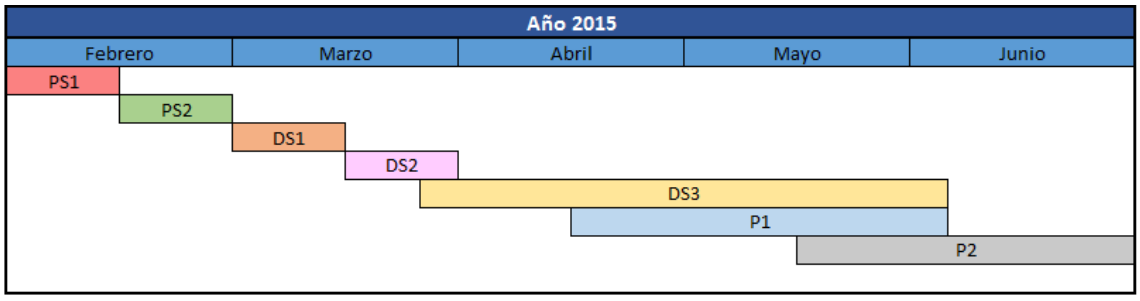
5.2. Plan de trabajo.

El proyecto se planeó para hacerlo a lo largo de cinco meses dedicándole una media de quince horas semanales. Gracias a una buena planificación se pudo terminar en el tiempo establecido.

A continuación se detalla la planificación temporal de cada una de las fases de la elaboración del proyecto.

Fase	Horas	Actividades
Planificación del sistema	50	Definición de requisitos (PS1)
		Definición de la arquitectura tecnológica (PS2)
Desarrollo del sistema	200	Análisis del sistema (DS1)
		Diseño del sistema (DS2)
		Construcción del sistema (DS3)
Pruebas	50	Pruebas de componentes internos del sistema (P1)
		Pruebas de uso (P2)

Y aquí como se distribuirán las diferentes actividades a lo largo de los meses.



6. Análisis.

El objetivo de esta fase es la obtención de una especificación detallada del sistema de información que satisfaga las necesidades de los usuarios y sirva de base para el posterior diseño del sistema.

6.1. Obtención de requisitos.

Ahora vamos a detallar las principales condiciones que debe cumplir el sistema de información:

- El sistema de información deberá eliminar los puntos de fallo únicos con la intención de ofrecer el servicio el mayor tiempo posible.
- El sistema de información se construirá con software libre.
- El sistema de información se construirá usando KVM como plataforma de virtualización.
- El sistema de información se construirá usando las soluciones de alta disponibilidad y balanceo de carga existentes para Linux CentOS.
- El sistema de información debe ser fácilmente escalable.
- El sistema de información debe ser seguro, sólo podrán acceder a él las personas autorizadas.
- El software de tienda virtual debe ser fácil de usar.
- El software de tienda virtual debe ser rápido.
- El software de tienda virtual debe contar con una cierta comunidad en español a la que poder recurrir en caso necesario.
- El software de tienda virtual debe proporcionar al administrador, como mínimo, las herramientas necesarias para gestionar el catálogo, pedidos, clientes y proveedores.
- El software de tienda virtual debe proporcionar al administrador ciertas estadísticas como número de visitas, número de personas que se han dado de alta, detalles sobre las ventas, información sobre el stock, etc.
- El software de tienda virtual debe permitir a los clientes registrarse.
- El software de tienda virtual debe permitir a los clientes modificar cualquier información facilitada al registrarse.
- El software de tienda virtual debe permitir a los clientes registrados identificarse con su correo electrónico y su contraseña.
- El software de tienda virtual debe permitir, sólo a los clientes que se han identificado, realizar compras.
- El software de tienda virtual debe permitir a los clientes pagar de forma segura.

6.2. Actores.

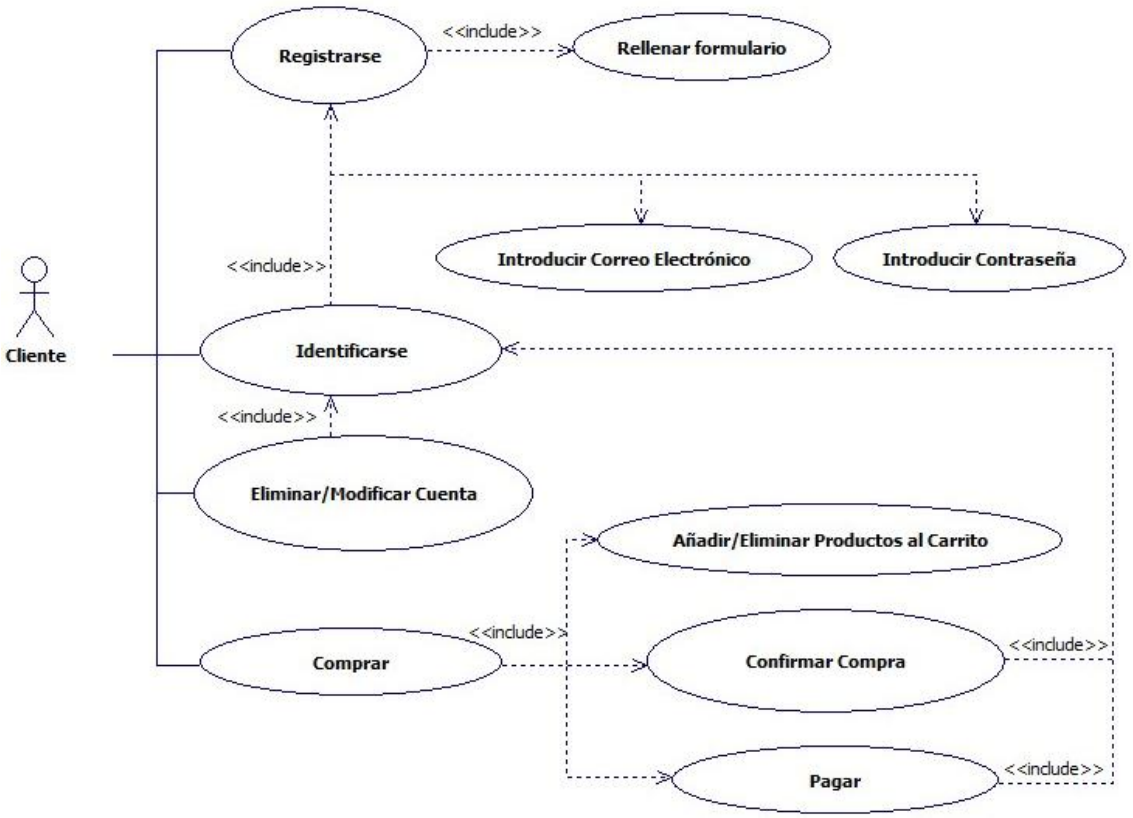
En la siguiente tabla se muestran los distintos actores que interactúan con el sistema y su descripción:

Actor	Descripción
Cliente	Usuario que accede a la tienda virtual para realizar sus compras.
Administrador	Usuario registrado responsable del funcionamiento de la tienda virtual.

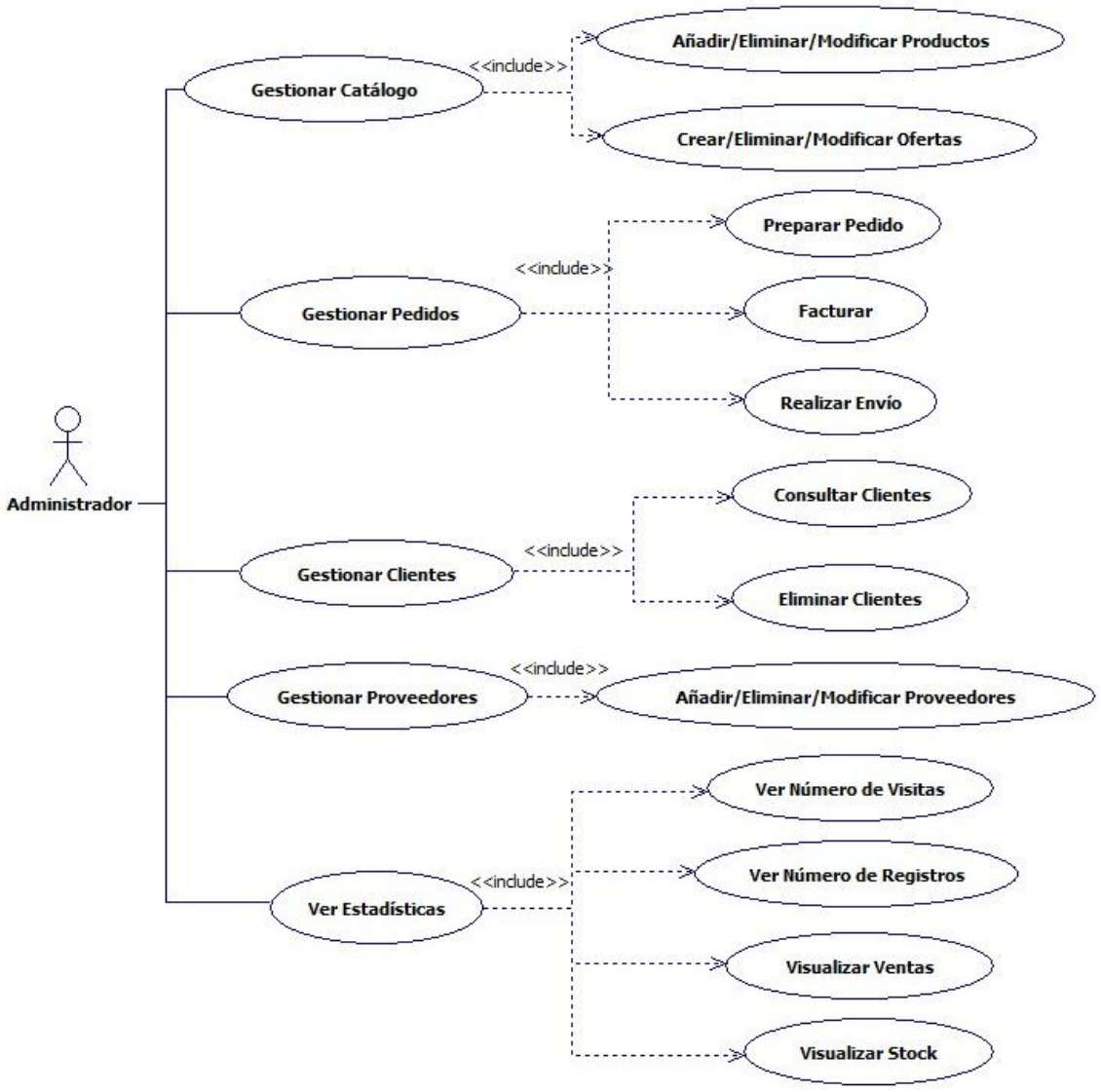
6.3. Casos de uso.

A continuación se describen los principales casos de uso que representan la forma en cómo un cliente o actor interactúa con el sistema que se va a desarrollar.

Rol	Caso de uso	Incluye	Incluye	Descripción
Cliente	Registrarse	Rellenar Formulario	-	Todos los clientes de la tienda virtual deben estar registrados en el sistema. Para ello debrán rellenar un formulario con sus datos personales
	Identificarse	Registrarse	-	Para identificarse es necesario introducir la contraseña y el correo electrónico que eligieron a la hora de registrarse
		Introducir Correo Electrónico	-	
		Introducir Contraseña	-	
	Eliminar/Modificar Cuenta	Identificarse	-	Un cliente puede modificar cualquiera de sus datos personales, incluso puede darse de baja, siempre y cuando esté identificado en el sistema
	Comprar	Añadir/Eliminar Productos al Carrito	-	Cualquier cliente puede ver la tienda virtual e ir añadiendo productos al carrito . Sólo aquellos que se han identificado pueden confirmar el pedido para finalmente realizar el pago.
		Confirmar Compra	Identificarse	
		Pagar	Identificarse	



Rol	Caso de uso	Incluye	Descripción
Administrador	Gestionar Catálogo	Añadir/Eliminar/Modificar Productos	El administrador se encarga de poner en la tienda los articulos que están a la venta con todas sus características: precio, categoría, stock, etc.
		Crear/Eliminar Ofertas	
	Gestionar Pedidos	Preparar Pedido	Desde que el cliente realiza el pago, se prepara la mercancía y se factura para su posterior envío
		Facturar	
		Realizar Envío	
	Gestionar Clientes	Consultar Clientes	El administrador puede acceder al listado de clientes, en cualquier momento puede eliminarlos
		Eliminar Clientes	
	Gestionar Proveedores	Añadir/Eliminar/Modificar Proveedores	El administrador puede dar de alta nuevos proveedores, modificar los datos de alguno existente o bien eliminarlo
	Ver Estadísticas	Ver Número de Visitas	El administrador puede ver información importante acerca del funcionamineto de la tienda virtual
		Ver Número de Registros	
		Visualizar Ventas	
		Visualizar Stock	



6.4. Análisis de las alternativas de balanceo de carga.

En este capítulo vamos a analizar las alternativas que existen a la hora de realizar el balanceo de carga y veremos las ventajas y desventajas que tienen para al final decantarnos por una.

Linux Virtual Server:

- Constituido por un servidor altamente escalable y de alta disponibilidad.
- Balance de carga mediante nodo dedicado con sistema operativo GNU/Linux.
- Clúster completamente transparente al usuario final.
- Mecanismo para que el clúster funcione con una IP pública.
- Permite el manejo de protocolos como HTTP, FTP, DNS, entre otros.

Piranha:

- Implementación de software de alta disponibilidad.
- Interfaz web para control completo del clúster.
- Posee herramientas para su monitorización.
- Balance mediante direcciones IP.
- Permite el manejo de protocolos como HTTP, FTP, DNS, entre otros.

Ultramonkey:

- Permite balanceo de carga y alta disponibilidad.
- Incluye monitorización de servidores reales y nodos balanceados.
- Permite el manejo de protocolos como HTTP, FTP, DNS, entre otros.
- Usado desde clústeres de dos nodos hasta grandes sistemas.

Conga:

- Sólo permite alta disponibilidad.
- Interfaz web para control completo del clúster.
- Fácil integración con los clústeres.
- Tiene paquetes para el almacenamiento compartido.

Software	Ventajas	Desventajas
LVS	• Fácil comprensión de funcionamiento. • Amplia gama deconfiguraciones. • Funciona a nivel TCP/IP.	• Algunos esquemas se ven afectados por la cantidad de servidores reales que se pueden utilizar. • Cuando el número de servidores reales es elevado se genera mucho tráfico en la red de trabajo.
Piranha	• Soporte por parte de RedHat Inc. • Fácil instalación debido al formato de distribución. • Administración y manejo mediante interfaz web. • Monitorización de servidores reales.	• Sólo disponible para versiones RedHat.
Ultramonkey	• Múltiples esquemas de configuración. • Varios formatos para su instalación. • Amplia documentación sobre cada componente. • Se apoya en el proyecto LVS . • No es dependiente de una distribución de GNU/Linux en particular.	• Los nodos directores tienen que ejecutar estrictamente el sistema operativo GNU/Linux. • La configuración puede resultar compleja según el esquema elegido
Conga	• Soporte por parte de RedHat Inc. • Fácil instalación debido al formato de distribución. • Administración y manejo mediante interfaz web. • Posee herramientas para usar almacenamiento compartido.	• Sólo disponible para versiones RedHat.

Después de haber analizado algunas de las herramientas disponibles para poder implementar el balanceo de carga y la alta disponibilidad del sistema de información, nos hemos decantado por las soluciones de RedHat: Conga para la creación de un clúster de bases de datos en alta disponibilidad y Piranha para la creación del clúster de servidores web con balanceo de carga.

6.5. Análisis de los modos de balanceo de carga.

Ahora vamos a analizar los métodos que usan los servidores virtuales creados con Piranha para balancear las peticiones de los usuarios.

- *Servidor virtual usando NAT:*

NAT (*N*etwork *A*ddress *T*ranslation) es una técnica utilizada para que una máquina reciba información dirigida a otra y ésta pueda reenviarla a quien la solicitó inicialmente. Para ello la máquina que recibe la información, en forma de paquetes, deberá reescribir los paquetes sustituyendo su propia dirección con la de la máquina que realizó la petición. Una vez reescrito el paquete de la forma correcta el balanceador se encargará de enviar los paquetes por la

interface adecuada para que le lleguen a quien procesará la petición. Cuando el servidor responda, lo hará al balanceador y éste reescribirá el paquete, otra vez, poniendo en los paquetes la dirección del cliente que solicitó la información.

- *Servidor virtual usando IP Tunneling:*

Utilizando *NAT* teníamos un cuello de botella en el balanceador ya que tiene que reescribir y distribuir los paquetes del cliente al servidor y viceversa. Utilizando *IP Tunneling* el balanceador únicamente tendrá que hacerse cargo de las peticiones de los clientes y enviarlas a los servidores, siendo estos mismos los que responderán a los clientes. De esta forma el balanceador de carga puede manejar más nodos, es decir el servicio es más escalable.

Una vez el balanceador de carga tiene el paquete determina si pertenece a uno de los servicios que tiene balanceados. De ser así encapsula el paquete en otro paquete y se lo envía al servidor de destino. Es este el que se encarga de responder al cliente directamente sin pasar por el balanceador.

El balanceador guarda una tabla de conexiones y cuando le llega un paquete determina si ya existe una conexión abierta y de ser así que servidor real es el que está sirviéndola para enviarle el paquete.

Los servidores deberán estar configurados para trabajar con *IP Tunneling (encapsulation)* ya que cuando el balanceador recibe un paquete para uno de los servidores este lo encapsula en un datagrama IP y lo manda a uno de los servidores. Cuando el servidor lo recibe tendrá que obtener el datagrama y responder directamente al cliente sin pasar por el balanceador con lo cual los servidores tendrán que estar conectados tanto al balanceador como a los clientes.

- *Servidor virtual usando Direct Routing:*

Al igual que en *IP Tunneling* el balanceador sólo gestionará las peticiones del cliente hacía el servidor con lo cual es una solución altamente escalable.

La dirección virtual (VIP) es compartida por el balanceador y los servidores. De esta manera el balanceador recibe las peticiones y las envía a los servidores que procesan las peticiones y dan servicio directamente a los clientes.

En esta solución es necesario que alguna la de las interfaces del balanceador y de los servidores estén en el mismo segmento físico de red ya que el balanceador de carga cambiará su dirección física, MAC, en la trama por la dirección física de uno de los servidores que tendrá un alias con la dirección VIP.

El balanceador guarda una tabla de conexiones y cuando le llega un paquete determina si ya existe una conexión abierta y de ser así que servidor real es el que está sirviéndola para enviarle el paquete.

Hemos estado haciendo referencia a cómo el balanceador distribuirá las peticiones entre los servidores, pero para que esta distribución sea efectiva ha de ser planificada de alguna forma. A la hora de compilar el núcleo en el balanceador tendremos que escoger que algoritmos vamos a utilizar para hacer el balanceo de carga.

Los algoritmos más interesantes son los siguientes:

- *Round-Robin:*

  Este algoritmo es el más simple y lo que hace es distribuir las peticiones de manera alternada entre los servidores. Esta distribución es muy sencilla pero presupone que todas las peticiones van a ser equivalentes, en términos de carga, para el servidor, algo que en la realidad dista mucho de ser cierto; o que la capacidad de procesamiento de los servidores es la misma.

- *Weighted Round-Robin:*

  Este algoritmo permite un aprovechamiento mejor del clúster cuando hay máquinas con diferentes capacidades de procesamiento, de esta forma a las máquinas con mayor capacidad de procesamiento se les dará una mayor prioridad para responder a las peticiones de los clientes y el balanceador distribuirá la carga entre los servidores teniendo en cuenta su prioridad.

- *Least-Connection:*

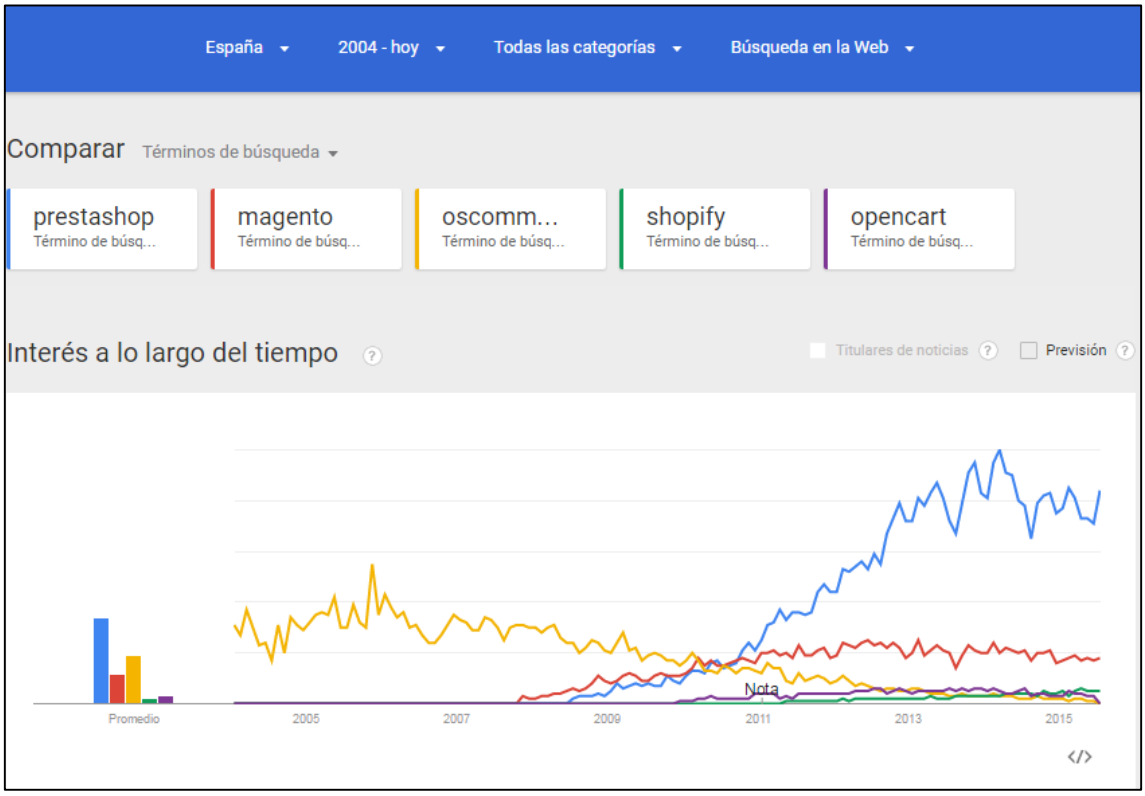
  Con este algoritmo las peticiones se enviarán al servidor que menos conexiones este sirviendo en ese momento. Si la capacidad de procesamiento de los servidores es similar este algoritmo distribuirá la carga de forma óptima entre todas las máquinas del clúster. Sin embargo si las capacidades de procesamiento varían mucho, la carga no será repartida de forma ecuánime ya que la carga se repartirá según el número de conexiones abiertas en ese momento y no sobre la carga real de cada máquina.

- *Weighted Least-Connection:*

  A cada servidor se le asigna una prioridad según su capacidad de procesamiento y aquellos con más prioridad atenderán más peticiones, es decir tendrán más conexiones abiertas.

6.6. Análisis de alternativas de tienda virtual.

En la actualidad existen cantidad de soluciones para crear nuestra tienda virtual. A continuación se usa la herramienta *"Trends"* de Google para conocer la tendencia de cinco de las más importantes.



Ahora vamos a realizar una comparativa para decidir cuál se ajusta mejor a las necesidades de los usuarios.

Software	Diseño	Velocidad	Usabilidad	Comunidad
Prestashop	Diseño profesional y muy actual	Alta	Fácil de usar	Bastante amplia. Sigue creciendo
Magento	Diseño visual muy atractivo	Baja	Se necesitan amplios conocimientos para poder usarlo	Pequeña, casi todo en ingés
Oscommerce	Diseño usando tablas	Media	Fácil de usar	Gran comunidad en inglés.
Shopify	Simplicidad, minimalismo y elegancia	Alta	Fácil de usar	Comunidad pequeña.
Opencart	Diseño visual muy atractivo	Alta	Fácil de usar	Comunidad pequeña

Se ha decidido que se instalará Prestashop debido a:

- Su diseño profesional y actual.
- Es uno de los más fáciles de usar.
- Es muy rápido, según el último estudio realizado por el equipo de Prestashop, los test de Google a día de hoy le dan como la aplicación ecommerce más rápida.
- Cuenta con una considerable comunidad en español.
- Es el software que más se usa en España y ha sido **g**alardonado como mejor aplicación para comercio electrónico en 2010 y 2011, consecutivamente, por lo que Prestashop es una apuesta segura.

Los requisitos necesarios para realizar su instalación son:

- Sistema operativo: Windows, Mac o Linux.
- Servidor web: Apache 1.3, Apache 2.x, Nginx o Microsoft IIS.
- PHP 5.2 o superior.
- MySQL 5.0 o superior instalado con una base de datos creada.

Opcionalmente:

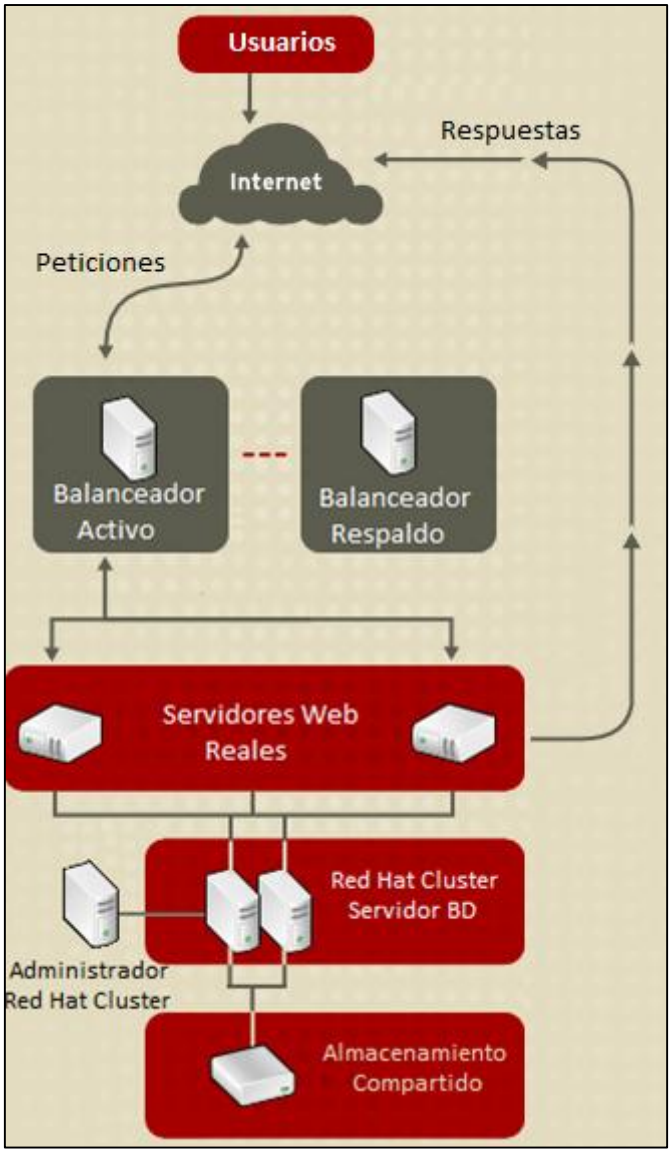
- En la configuración PHP
  - memory_limit = "64M".
  - file_max_upload_size = "16M".

- Extensiones PHP opcionales:
  - GD.
  - cURL.
  - SimpleXML.
  - SOAP.

7. Diseño.

El objetivo de esta etapa es la especificación de la arquitectura del sistema y del entorno tecnológico que le va a dar soporte, junto con la definición detallada de los componentes del sistema de información.

7.1. Diseño de la topología del clúster.



El sistema de información estará formado por siete máquinas virtuales:

- TFG_AdminClusterBD: Administrador del clúster de base de datos en alta disponibilidad.
- TFG_ServerBD1: Servidor de base de datos activo.
- TFG_ServerBD2: Servidor de base de datos de respaldo.
- TFG_LoadBalancer1: Balanceador activo.
- TFG_LoadBalancer2: Balanceador de respaldo.
- TFG_WebServer1: Servidor web.
- TFG_WebServer2: Servidor web.

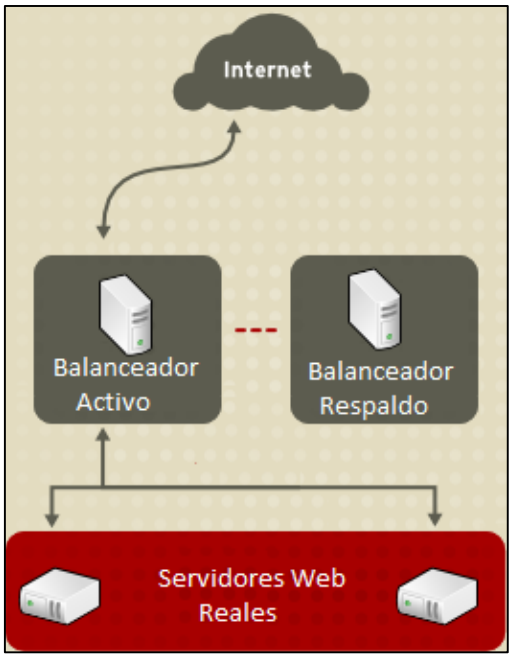
Los usuarios a través de Internet accederán al servicio de tienda virtual Prestashop, usando para ello la IP virtual del servicio. Estas peticiones serán recibidas por el balanceador activo que se encargará de repartir las peticiones según la cantidad de trabajo que tengan los servidores web. En caso de que este balanceador falle, existe uno de respaldo que se encargará de asumir sus funciones. Los servidores web atacarán al servidor de base de datos activo para acceder a la información que estará almacenada en la base de datos alojada en la cabina de discos. Existe un servidor de base de datos de respaldo que seguirá ofreciendo el servicio cuando el activo falle.

7.2. Identificación de los subsistemas de diseño.

El sistema se ha estructurado en dos subsistemas más pequeños y simples. La razón de esta estructuración es que de esta manera se simplifica la construcción, prueba y mantenimiento del sistema.

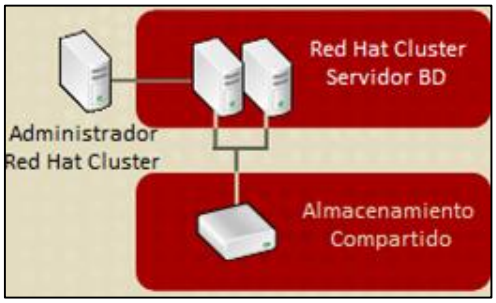
Por un lado tendremos el clúster de servidores web con balanceo de carga. Este subsistema lo compondrán las siguientes máquinas:

- TFG_LoadBalancer1
- TFG_LoadBalancer2
- TFG_WebServer1
- TFG_WebServer2



Por otro lado está el clúster de servidores de base de datos en alta disponibilidad compuesto por:

- TFG_AdminClusterBD
- TFG_ServerBD1
- TFG_ServerBD2



7.3. Diseño de la infraestructura de red.

El objetivo de esta tarea es identificar y diseñar la red de comunicaciones del sistema de información, la cual está pensada para que sea lo más simple posible. Serán necesarias tres redes distintas:

- Red aislada TFG_RedAislada0 (10.22.122.0): Necesaria para las comunicaciones del clúster de alta disponibilidad.
- Red del laboratorio (10.22.146.0): Necesaria para que los servidores de base de datos puedan acceder a la cabina de discos. Implica poner la interfaz de red en modo bridge.
- Red aislada NAT (192.168.122): Necesaria para comunicar todas las máquinas. Necesita que el sistema anfitrión tenga activado el ip_forward e implica poner la interfaz de red en modo NAT.

En la siguiente imagen se muestran las IPs de las distintas máquinas:

Nombre de la MV	Interfaz de red virtual	IP
TFG_AdminClusterBD	NAT	DHCP
	Red aislada	Estática 10.22.122.10
TFG_ServerBD1	NAT	DHCP
	Red aislada	Estática 10.22.122.11
	Bridge	DHCP 10.22.146.202
TFG_ServerBD2	NAT	DHCP
	Red aislada	Estática 10.22.122.12
	Bridge	DHCP 10.22.146.203
TFG_LoadBalancer1	NAT	Estática 192.168.122.11
TFG_LoadBalancer2	NAT	Estática 192.168.122.12
TFG_WebServer1	NAT	Estática 192.168.122.21
TFG_WebServer2	NAT	Estática192.168.122.22

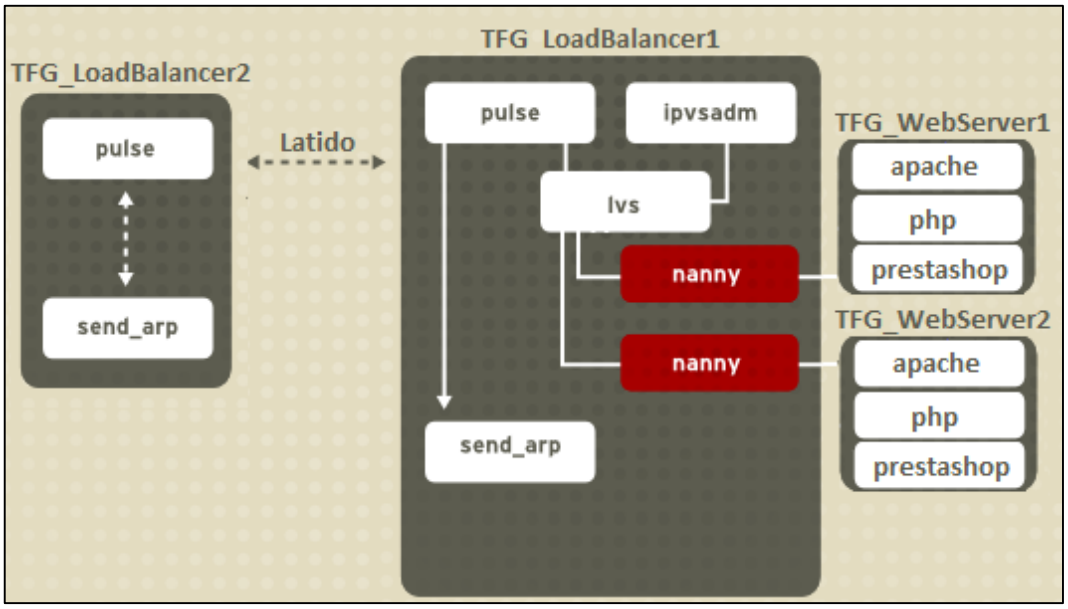
Además existen otras IPs importantes:

- 10.22.146.190: Dirección de la cabina de discos
- 192.168.122.50: IP virtual del servicio de tienda virtual.
- 192.168.122.100: IP virtual de servidor de base de datos.

7.4. Diseño de la arquitectura de servicios.

Esta tarea consiste en la identificación de los principales servicios que componen la solución elegida.

7.4.1. Clúster de servidores web con balanceo de carga.



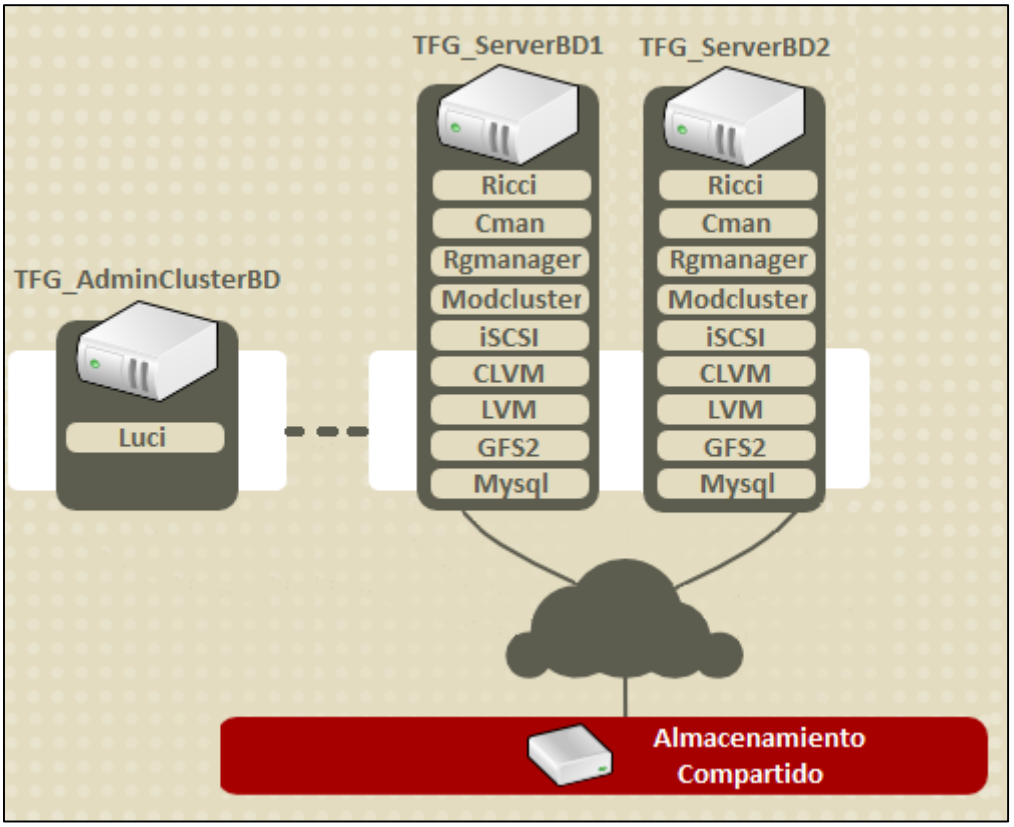
Seguidamente se explicarán cómo se lleva a cabo el balanceo de carga.

El servicio ***pulse*** se ejecutará tanto en el balanceador LVS (Linux Virtual Server) activo como en el de respaldo. En el de respaldo, este demonio se encargará de enviar una señal "latido" a la interfaz pública del balanceador activo para asegurarse de que éste esté funcionando correctamente. En caso de que el LVS de respaldo no reciba ninguna respuesta a los latidos emitidos, éste iniciará un proceso de recuperación contra fallos llamando a ***send_arp*** para que asigne la IP virtual a su MAC, envía una orden para apagar el demonio ***lvs*** activo e inicia el suyo propio para que acepte solicitudes para los servidores virtuales configurados.

En el balanceador activo, ***pulse*** tendrá la tarea de iniciar el demonio ***lvs*** y la de responder a los latidos que se le han enviado. Una vez iniciado el demonio ***lvs***, éste usará la utilidad ***ipvsadm*** para configurar y mantener la tabla de rutas IPVS (IP Virtual Server) en el kernel y además iniciará un proceso ***nanny*** para cada servidor real que haya configurado en el servidor virtual. Los procesos ***nanny*** revisarán el estado de los servidores reales e informarán al demonio ***lvs*** si el servicio está o no está funcionando. Si el servicio no está funcionando, el demonio ***lvs*** ordena a ***ipvsadm*** que borra el servidor real de la tabla de rutas IPVS.

En cuanto a los servidores reales, éstos ofrecerán el servicio de tienda virtual Prestashop. Para ello será necesario instalar en ambos el servidor web Apache y PHP.

7.4.2. Clúster de servidores de base de datos en alta disponibilidad.



Como se puede observar en la figura, en el nodo administración se ejecutará el servidor luci. Este servicio posee la interfaz gráfica del Conga (Conjunto de software desarrollado para proporcionar el software necesario para la creación y manejo de clústeres). Desde él se realizarán las actividades de creación y administración del clúster.

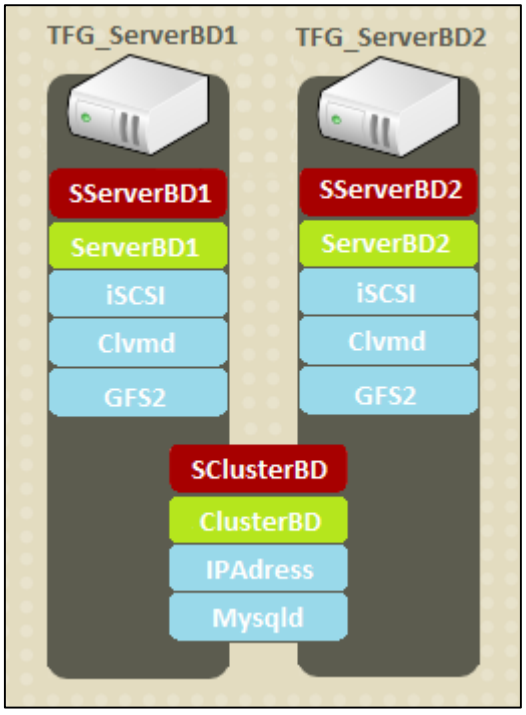
En los nodos que forman el clúster, se necesitará el resto de componentes del Conga:

- Ricci: Encargado de propagar la información del clúster actualizada a todos los nodos que lo componen.
- Cman: Responsable del manejo del clúster.
- Rgmanger: Encargado del manejo de los grupos de servicios del clúster.
- Modcluster.

En cuanto al almacenamiento compartido, su usará iSCSI para acceder al volumen que se ha creado en la cabina. Tras esto, los nodos verán como suyo el espacio que se les ha asignado, pero todavía no lo podrán utilizar. En el disco que se ha exportado habrá que crear un volumen lógico para que lo utilice el clúster. CLVM será el encargado del manejo de este tipo de volúmenes y lvm sincronizará el acceso a dicho volumen. Ahora sólo hace falta crear un sistema de archivos distribuido mediante GFS2. Todo esto se conoce como almacenamiento persistente (Resilient Storage).

Finalmente nos hará falta el software del gestor de base de datos Mysql para que acceda a la base de datos ubicada en el almacenamiento compartido y crear la infraestructura necesaria para ponerlo en alta disponibilidad. Para ello, una vez creado el clúster, tendremos que:

- *Crear los recursos que serán manejados por él*: IPAdress, Mysql, iSCSI, Clvmd y GFS2.
- *Crear los failover*: El dominio failover restringirá en que nodos se puede ejecutar un determinado grupo de servicios. Se creará un failover por cada nodo del clúster y uno más para el servicio de alta disponibilidad.
    - ServerBD1: Se ejecutará en el nodo TFG_ServerBD1.
    - ServerBD2: Se ejecutará en el nodo TFG_ServerBD2.
    - ClusterBD: Se ejecutará en ambos nodos.
- *Crear los grupos de servicio:* Un grupo de servicios es un conjunto de recursos que se activarán en unos determinados nodos especificados por el failover.
    - SServerBD1: Usará el failover ServerBD1 y los recursos iSCSI, Clvmd y GFS2.
    - SServerBD2: Usará el failover ServerBD2 y los recursos iSCSI, Clvmd y GFS2.
    - SClusterBD: Usará el failover ClusterBD y los recursos IPAdress y Mysqld.



8. Requisitos hardware y software.

Ahora que ya se ha realizado el análisis y diseño de la solución, estamos en condiciones de indicar que requisitos hardware y software son necesarios para llevar a cabo el despliegue del sistema información.

8.1. Requisitos hardware.

El hardware que se ha utilizado para el desarrollo de este trabajo de fin de grado es:

- Un ordenador personal en uno de los laboratorios de investigación con acceso al sistema anfitrión.
- Sistema anfitrión capaz de albergar siete máquinas virtuales y que tenga acceso a la cabina de discos e Internet.
- Volumen creado en la cabina de discos.

En cuanto a las máquinas virtuales, éstas tendrán los siguientes recursos virtuales:

- Un procesador virtual.
- 1 Gb de memoria RAM.
- De una a tres interfaces de red según la máquina que sea.

Finalmente se muestran los recursos virtuales que son compartidos por varios nodos y que habrá que crear:

- 192.168.122.50: IP virtual para acceder al servicio web.
- 192.168.122.100: IP virtual para acceder al servidor de base de datos.
- TFG_RedAislada0: Red aislada para las comunicaciones entre los distintos nodos del clúster de base de datos en alta disponibilidad.

8.2. Requisitos software.

En cuanto al software, las exigencias que debe cumplir el sistema son:

8.2.1. Sistema anfitrión.

- Linux CentOS: Sistema operativo usado. Se realizará una instalación de escritorio.
- Servidor de escritorio remoto TigerVNC: Necesario para que el ordenador del laboratorio pueda acceder al sistema anfitrión.
- Plataforma de virtualización KVM. Implica todo el software necesario para la creación de las máquinas y recursos virtuales.
- SSH: Programa que se usará para acceder a las máquinas virtuales.
- Gedit: Editor de textos.
- Mozilla Firefox: Navegador web que se usará para acceder a las interfaces gráficas para la creación y configuración de los clústeres, así como para la instalación del servicio web.
- Sysctl: Archivo para modificar parámetros del kernel. Se activará el ip_forwarding para permitir el enrutamiento mediante NAT.
- Selinux: Módulo de seguridad que habrá que desactivar.

8.2.2. Ordenador del laboratorio.

- Cliente de escritorio remoto TigerVNC: Necesario para conectarse con el sistema anfitrión.
- SSH: Programa que se usará para acceder al sistema anfitrión de forma remota en caso de que TigerVNC no funcione.

8.2.3. Máquinas virtuales.

Antes de nada vamos a exponer el software que van a tener en común todas las máquinas virtuales.

- Linux CentOS 6.4: Sistema operativo usado. Se realizará una instalación mínima.
- Iptables: Herramienta para el filtrado de paquetes.
- Selinux: Módulo de seguridad que desde la página de Red Hat aconsejan deshabilitar si se usa junto a GFS2.
- Vi: Editor de textos.
- Scp: Protocolo de transferencia segura de archivos.
- Wget: Aplicación que permite la descarga de contenido desde servidores web.
- Unzip: Compresor/Descompresor de archivos en formato zip.

Ahora nos centraremos en conocer el software específico para cada tipo de máquina virtual.

Balanceadores:

- Piranha-gui: Interfaz gráfica de instalación y administración del servidor virtual de Linux (LVS).
- Ipvsadm: Herramienta de línea de comandos utilizada para la instalación y administración del LVS.
- Pulse: Servicio encargado de monitorear el estado de un LVS.
- Sysctl: Archivo para modificar parámetros del kernel. Se activará el ip_forwarding para que los balanceadores sean capaces de reenviar las peticiones a los servidores web.

Servidores web:

- Apache: Servidor web HTTP de código abierto.
- PHP: Lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico.
- Prestashop: Servicio web de tienda virtual.
- Arptables: Herramienta que nos ayuda a administrar las reglas de filtrados de los paquetes ARP.
- Sysctl: Archivo para modificar parámetros del kernel. Se deshabilitarán los parámetros arp_announce/arp_ignore para que funcione correctamente el balanceo de carga.

Servidores de base de datos.

- Mysql: Sistema de gestión de bases de datos relacional, multihilo y multiusuario.
- Ricci: Componente del conga. Encargado de propagar la información del clúster actualizada a todos los nodos que lo componen.
- Cman: Responsable del manejo del clúster.
- Rgmanger: Componente del Conga. Encargado del manejo de los grupos de servicios del clúster.
- Modcluster: Componente del Conga.
- iSCSI: Programa que hace posible que los nodos puedan usar el volumen exportado de la cabina de discos.
- Clvm: Manejador de volúmenes lógicos del clúster.
- Dlm: Gestor de cerrojo distribuido encargado del acceso sincronizado al almacenamiento compartido.
- GFS2: Sistema de archivos distribuido que tendrá el volumen de la cabina.

Administrador del clúster de servidores de base de datos en alta disponibilidad.

- Luci: Componenete del Conga. Es la interfaz gráfica para la creación y administración del clúster de alta disponibilidad.

9. Construcción.

Una vez finalizadas las etapas de análisis y diseño tenemos claro cómo va a ser nuestro sistema de información, así que ya podemos empezar a construir los distintos componentes que lo conforman.

9.1. Preparación del sistema anfitrión y del puesto de trabajo.

Antes que nada, vamos a ver el software que está instalado en el sistema anfitrión y la configuración de los parámetros que son necesarios.

- Plataforma de virtualización KVM.

```
$ yum  groupinstall Virtualization
$ yum groupinstall “Virtualization Client”
$ yum groupinstall “Virtualization Platform”
$ yum groupinstall  “Virtualization Tools”
```

- Se debe habilitar el IP forwarding para poder poner las interfaces en modo bridge.

```
$ vi /etc/sysctl
```

Modificamos el valor de la variable “*ip_forward”.*

```
net.ipv4.ip_forward = 1
```

Y hacemos que el cambio tenga efecto.

```
$ sysctl -p /etc/sysctl.conf
```

- Se debe desactivar SELinux.

```
$ vi /etc/selinux/config
```

Modificamos el valor de la variable “*SELINUX”.*

```
SELINUX = disabled
```

Y reiniciamos la máquina.

```
$ reboot
```

En cuanto al software utilizado para conectarse desde el equipo del laboratorio con el sistema anfitrión, se ha utilizado la aplicación de escritorio remoto TigerVNC.

- En el sistema anfitrión estaba instalado el servidor.

```
$ yum install tigervnc-server
```

- En el puesto de trabajo, tras haber instalado el sistema operativo Linux CentOS 6.4 y haber configurado la red, se instaló el cliente.

```
$ yum install tigervnc-server
```

9.2. Creación de los recursos hardware.

En este apartado vamos a ver cómo se crean los recursos necesarios que nos permitirán crear los clústeres para desplegar el servicio de tienda virtual.

9.2.1. Creación de las máquinas virtuales.

Vamos a explicar cómo se crea una máquina virtual y luego veremos cómo clonarla para facilitar el proceso de creación de todas las máquinas virtuales.

9.2.1.1. Creación de la primera máquina virtual.

```
$ virt-install --name=TFG_AdminClusterBD \
--vcpus=1 --ram=1024  \
--disk path=/mvirt/egarcia/TFG/AdminClusterBD.img,size=10 \
--location=/ImagenesDistroISO/centos/CentOS-6.4-x86_64-bin-DVD1.iso \
--os-type=linux \
--os-variant=rhel6
```

Donde:

--name: Nombre de la máquina virtual.

--vcpus: Nº de CPUs virtuales que usa.

--ram: Cantidad de memoria que tiene asignada.

--disk path,size: Lugar donde se guardará la imagen del disco y su tamaño en MB.

--location: Lugar donde se encuentra la imagen del sistema operativo que se instalará.

--os-type: Tipo de sistema operativo.

--os-variant: Tipo de distribución del sistema operativo.

9.2.1.2. Instalación del sistema operativo.

Tras ejecutar la anterior orden se abrirá una pantalla para empezar con la instalación del sistema operativo de la máquina virtual.

Durante la instalación:

- Selección del lenguaje: español.
- Selección del teclado: español.
- Tipos de dispositivos: dispositivos de almacenamiento básicos.
- Indicamos que se descarten todos los datos del dispositivo de almacenamiento.
- Nombre del host: localhost.localdomain.
- Huso horario: Atlántico/Canarias.
- Indicamos cual va a ser la contraseña del root.
- Creamos un diseño personalizado como el siguiente:

- Dejamos que instale en /dev/vda el gestor de arranque.
- Hacemos una instalación mínima (Minimal).

Una vez terminada la instalación de todos los paquetes ya tenemos la primera máquina virtual funcionando.

9.2.1.3. Clonación de las restantes máquinas virtuales.

Ahora que ya tenemos una máquina virtual creada y funcionando, en vez de repetir el proceso de creación con su correspondiente instalación del SO, clonaremos la máquina anterior con el fin de agilizar la obtención de todas las máquinas necesarias.

Se va a explicar cómo realizar la clonación de una de las máquinas y luego habrá que repetir el proceso teniendo en cuenta la siguiente tabla:

Nombre de la MV	Localización de la imagen del disco duro
TFG_AdminClusterBD	/mvirt/egarcia/TFG/AdminClusterBD
TFG_ServerBD1	/mvirt/egarcia/TFG/ServerBD1
TFG_ServerBD2	/mvirt/egarcia/TFG/ServerBD2
TFG_LoadBalancer1	/mvirt/egarcia/TFG_Balanceo/LoadBalancer1
TFG_LoadBalancer2	/mvirt/egarcia/TFG_Balanceo/LoadBalancer2
TFG_WebServer1	/mvirt/egarcia/TFG_Balanceo/WebServer1
TFG_WebServer2	/mvirt/egarcia/TFG_Balanceo/WebServer2

Lo primero que vamos a hacer es copiar el archivo xml que describe la máquina.

```
$ cd /etc/libvirt/qemu

$ cp TFG_AdminClusterBD.xml TFG_ServerBD1.xml
```

Luego copiamos la imagen del disco.

```
$ cp /mvirt/egarcia/TFG/AdminClusterBD.img /mvirt/egarcia/TFG/ServerBD1.img
```

A continuación editamos el archivo que acabamos de copiar.

```
$ vi TFG_ServerBD1.xml
```

Y nos centramos en las líneas:

```
<uuid>c11976df-40b6-1a37-ded0-192c544971c9</uuid>
<name>TFG_AdminClusterBD</name>
<source file='/mvirt/egarcia/AdminClusterBD.img'/>
<interface type='network'>
      <mac address='52:54:00:2a:b7:e1'/>
      <source network='default'/>
      <model type='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>
</interface>
```

Con el objetivo de:

- Elimminar el *"uuid"* para que sea el propio software de virtualización quien genere un uuid único.
- Modificar el *"name"* para poner el nombre de la nueva máquina virtual.
- Modificar el *"source file"* para añadir su correspondiente imagen del disco.
- Modificar la *"mac"* de la interfaz de red para poner una nueva que se genera usando el script que se adjunta en el capítulo Anexos.
- Eliminar el *"addres type"* para que sea el propio software de virtualización quien le asigne un slot disponible.

Quedando el archivo de la siguiente manera:

```
<name>TFG_ServerBD1</name>
<source file='/mvirt/egarcia/ServerBD1.img'/>
<interface type='network'>
      <mac address='00:16:3e:5a:4a:8f'/>
      <source network='default'/>
      <model type='virtio'/>
</interface>
```

Finalmente guardamos los cambios y definimos la nueva máquina virtual.

```
$ virsh define TFG_ServerBD1.xml
```

Una vez terminado el anterior paso, en el administrador de máquinas virtuales, aparecerá la nueva que acabamos de crear.

Una cosa a tener en cuenta después de haber clonado las máquinas, es que la nueva tarjeta de red no tiene el nombre esperado tal y como se evidencia al ejecutar la siguiente orden:

```
$ ifconfig -a
```

Esto se debe a que al clonar la máquina y modificar la MAC de la interfaz que tenía, en el archivo /etc/udev/rules.d/70-persistent-net.rules aparecen las MAC de dos interfaces, la antigua que ya no existe (eth0) y la nueva que acabamos de modificar (eth1).

Esta situación se podría solucionar teniendo en cuenta cuales son las interfaces activas a la hora de realizar su configuración. Una opción mucho mejor es eliminar el archivo /etc/udev/rules.d/70-persistent-net.rules y luego reiniciar la máquina para que lo vuelva a crear.

```
$ rm -rf /etc/udev/rules.d/70-persistent-net.rules

$ reboot
```

Al hacer esto, en el nuevo archivo sólo aparece la interfaz que realmente está conectada, pero se ha quedado el nombre de la interfaz tal y como estaba antes de haberlo borrado. Así que editaremos el archivo y pondremos el nombre correcto, eth0. Después de esto reiniciamos la máquina y ya podemos empezar a configurar las interfaces.

9.2.2. Creación de una red virtual aislada.

Ahora vamos a explicar cómo crear una red virtual aislada, que será la que use el clúster de bases de datos en alta disponibilidad. Para ello nos dirigimos al directorio /etc/libvirt/qemu/networks y creamos el correspondiente archivo de configuración en formato xml con la siguiente información.

```
$ vi TFG_RVirt0.xml
```

```
<network>
  <name>TFG_RVirt0</name>
  <uuid>a842d148-5fad-1e91-04d7-7b37d8356142</uuid>
  <bridge name='TFG_RVirt0' stp='on' delay='0' />
  <mac address='52:54:00:35:01:5D'/>
  <ip address='10.22.122.1' netmask='255.255.255.0'>
  </ip>
</network>
```

A continuación definimos la red aislada y la configuramos para que se inicie siempre de forma automática.

```
$ virsh net-define TFG_RVirt0.xml

$ virsh net-autostart TFG_RdVirtual0.xml
```

9.2.3. Creación y configuración de las interfaces de red.

En este momento ya tenemos todos los elementos necesarios para añadir todas la interfaces en las máquinas que necesiten más de una interfaz, usando para ello el administrador de máquinas virtuales. Para generar las MAC se usa el script que se adjunta en el apartado Anexos.

A continuación se muestra la tabla con las interfaces que necesita cada máquina y con los datos necesarios para realizar su configuración:

Nombre de la MV	Interfaz	Modo	MAC	IP
TFG_AdminClusterBD	eth0	NAT	00:16:3e:e9:1b:dc	DHCP
	eth1	Bridge	00:16:3e:ca:93:49	Estática 10.22.122.10
TFG_ServerBD1	eth0	NAT	00:16:3e:5a:4a:8f	DHCP
	eth1	Bridge	00:16:3e:a6:50:70	Estática 10.22.122.11
	eth2	TFG_RVirt0	00:16:3e:e7:0d:79	DHCP 10.22.146.202
TFG_ServerBD2	eth0	NAT	00:16:3e:51:a4:18	DHCP
	eth1	Bridge	00:16:3e:59:dc:3f	Estática 10.22.122.12
	eth2	TFG_RVirt0	00:16:3e:8c:ff:11	DHCP 10.22.146.203
TFG_LoadBalancer1	eth0	NAT	00:16:3e:bc:5f:ec	Estática 192.168.122.11
TFG_LoadBalancer2	eth0	NAT	00:16:3e:39:cc:32	Estática 192.168.122.12
TFG_WebServer1	eth0	NAT	00:16:3e:c8:09:e1	Estática 192.168.122.21
TFG_WebServer2	eth0	NAT	00:16:3e:f5:fe:29	Estática 192.168.122.22

Existen dos tipos distintos de archivos que se usan para la configuración de todas las interfaces y que se ubicarán en la carpeta /etc/sysconfig/network-scripts/ifcfg-X, siendo X el nombre del dispositivo.

- Configuración de la interfaz para asignación de IP mediante DHCP.

```
DEVICE=eth0
HWADDR=00:16:3e:e9:1b:dc
TYPE=Ethernet
ONBOOT=yes
BOOTPROTO=dhcp
```

- Configuración de la interfaz para asignación de IP estática.

```
DEVICE=eth1
IPADDR=10.22.122.2
HWADDR=00:16:3e:ca:93:49
NETMASK=255.255.255.0
TYPE=Ethernet
ONBOOT=yes
BOOTPROTO=static
```

Para que la configuración de red tenga efecto sólo tenemos que reiniciar el servicio.

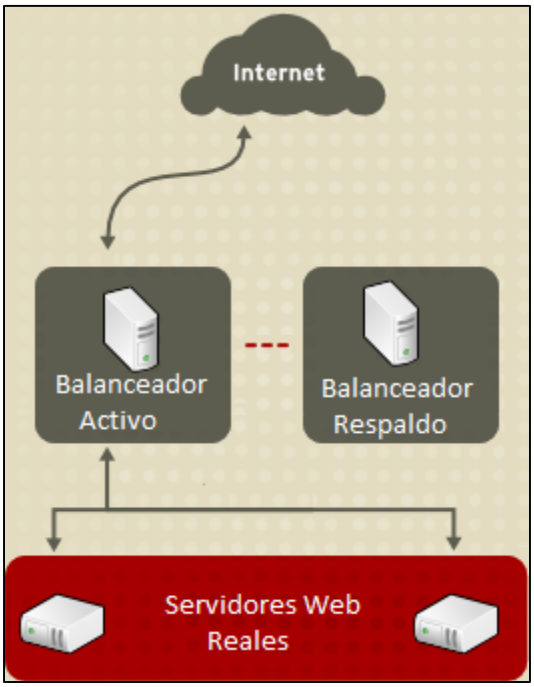
```
$ service network restart
```

9.2.4. Creación de un volumen en la cabina de discos.

Para terminar con los recursos necesarios, vamos a crear un volumen en la cabina de discos. Para ello accedemos a través de la interfaz de gestión de la cabina. Una vez dentro nos movemos por las distintas opciones hasta encontrar la que permite crear un volumen de 8 GB al que llamaremos *"tfg"* que se exportará usando iSCSI.

9.3. Creación y configuración del clúster de servidores web con balanceo de carga.

En este apartado se describirán los pasos que hay que realizar para hacer que las peticiones de los usuarios se repartan entre los dos servidores web.



Las máquinas con las que trabajaremos son:

- TFG_LoadBalancer1: Balanceador activo.
- TFG_ LoadBalancer2: Balanceador de respaldo.
- TFG_WebServer1: Servidor web activo.
- TFG_WebServer2: Servidor web activo.

9.3.1. Creación y configuración de los balanceadores.

Empezamos instalando el software que proporciona el balanceo de carga.

```
$ yum groupinstall "Load Balancer"
```

A continuación se configuran los servicios que son necesarios de forma que se arranquen cada vez que se enciendan las máquinas.

```
$ chkconfig pulse on

$ service pulse start

$ chkconfig piranha-gui on

$ service  piranha-gui start
```

Y creamos la contraseña del usuario "*piranha"* que usaremos para acceder a la interfaz gráfica.

```
$ piranha-passwd
```

Luego abrimos los puertos necesarios.

- Para httpd (Servidor web).

```
$ iptables -I INPUT -m state --state NEW -p tcp  --dport 80 -j ACCEPT
```

- Para piranha-gui (Interfaz gráfica para la configuración del LVS).

```
$ iptables -I INPUT -m state --state NEW -p tcp --dport 3636 -j ACCEPT
```

- Para pulse (Encargado de monitorear el estado de los LVS).

```
$ iptables -I INPUT -m state --state NEW -p tcp --dport 539 -j ACCEPT
```

- Para evitar que las conexiones se queden permanentemente activas.

```
$ iptables -I INPUT -m state --state INVALID -s 192.168.122.0/24 -j ACCEPT
```

Después activamos el reenvío de paquetes (IP Forwarding). Para ello accedemos al fichero /etc/sysctl.

```
$ vi /etc/sysctl
```

Modificamos el valor de la variable "*ip_forward".*

```
net.ipv4.ip_forward = 1
```

Y hacemos que el cambio tenga efecto:

```
$ sysctl -p /etc/sysctl.conf
```

Por último, también es necesario deshabilitar SELinux. Se accede al fichero /etc/selinux/config.

```
$ vi /etc/selinux/config
```

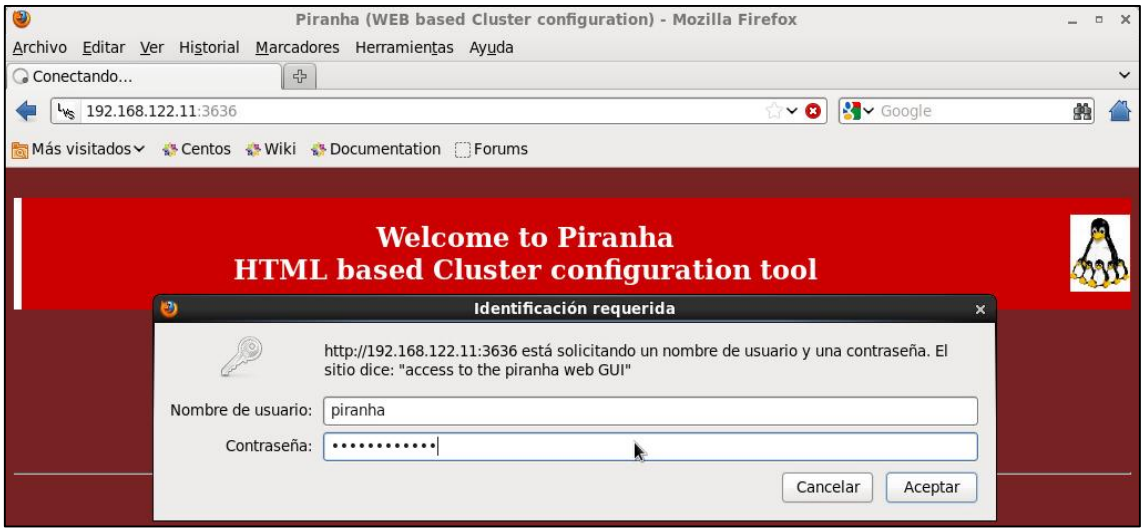
Modificamos el valor de la variable "*SELINUX".*

```
SELINUX = disabled
```

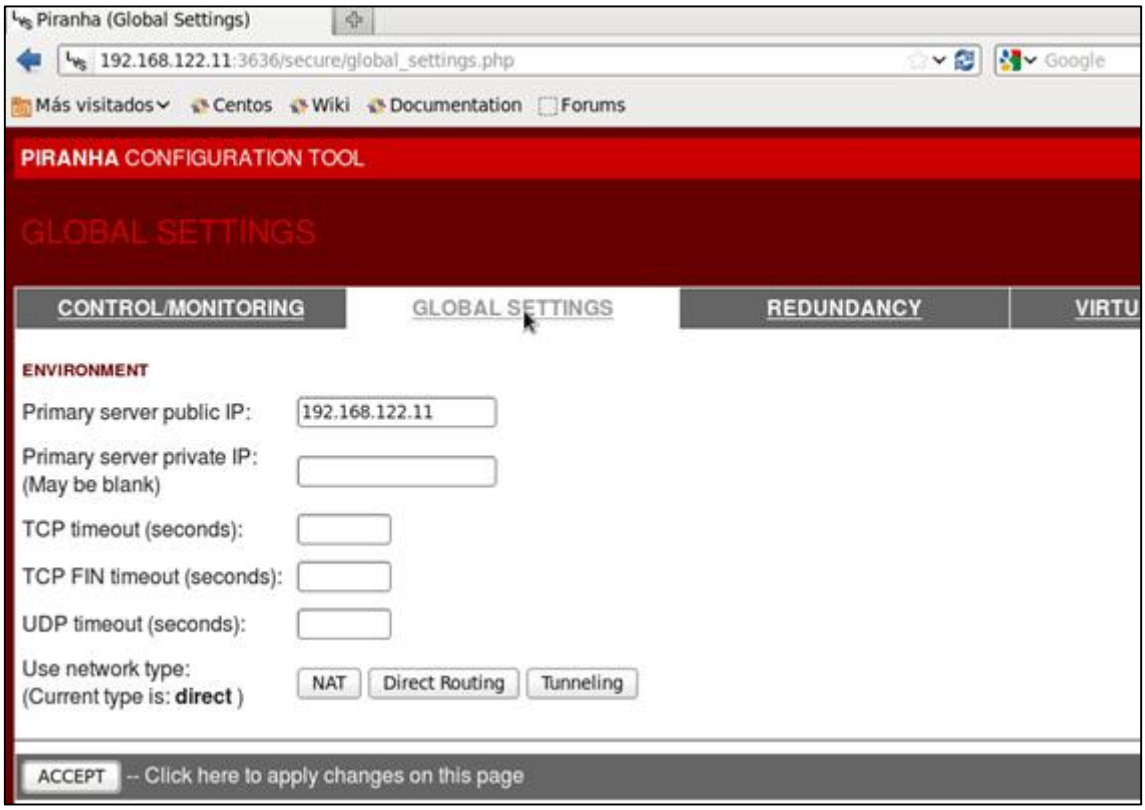
Y reiniciamos las máquinas.

```
$ reboot
```

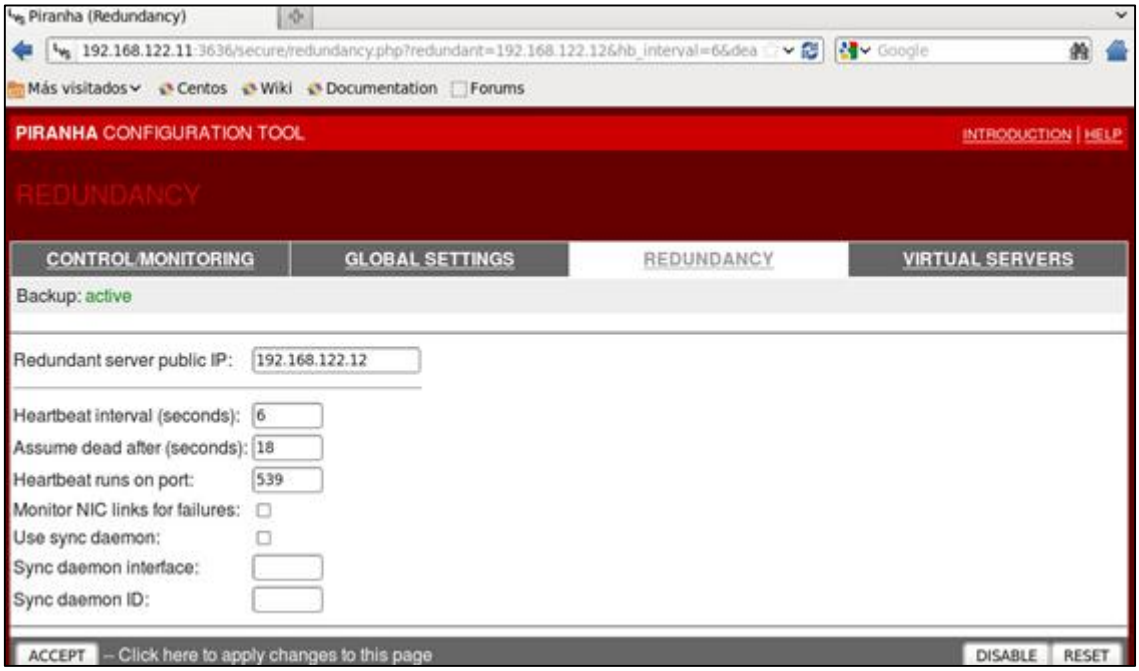
Ahora ya podemos acceder a la interfaz gráfica *"Piranha"* para empezar con la creación de la infraestructura que dará soporte al servicio con balanceo de carga. Para ello sólo tenemos que poner en el navegador web del sistema anfitrión la IP del balanceador principal seguida del puerto, es decir 192.168.122.11:3636.



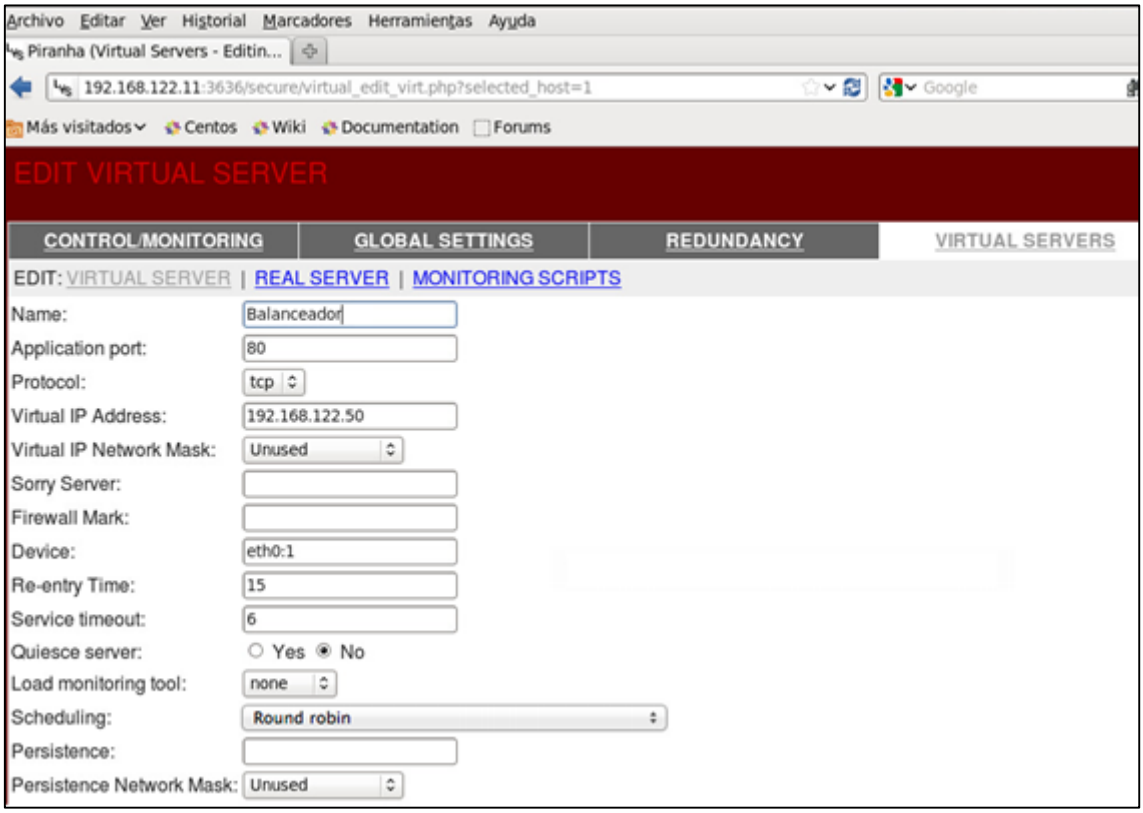
Empezamos configurando el balanceador activo cuya IP es la 192.168.122.11. Como ya se ha mencionado con anterioridad se elige la opción *"Direct Routing"* porque proporciona mayor rendimiento que el resto de configuraciones. Este punto es crucial, y cambiará toda la configuración de la solución final.



En esta otra pestaña se configura el balanceador de respaldo. Como se puede observar sólo es necesario añadir la IP de la máquina destinada a tal fin, la 192.168.122.12.

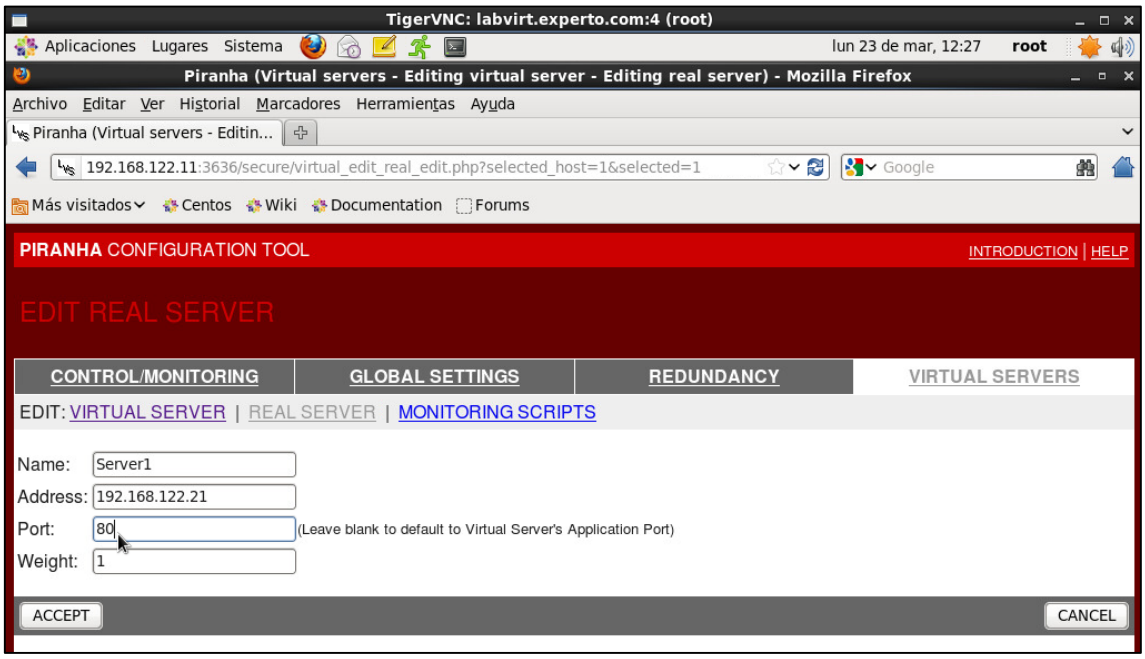


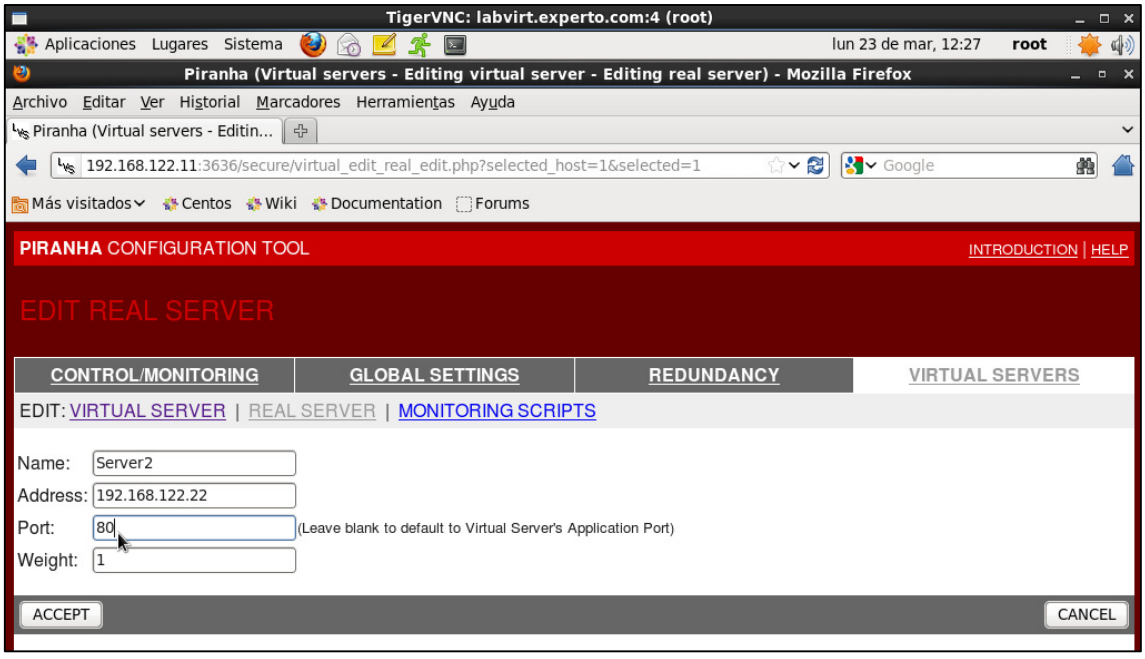
Ahora vamos a configurar un servidor virtual que con la IP virtual 192.168.122.50 que prestará el servicio. Las peticiones web (puerto 80 tcp) se van a balancear usando el algoritmo "*Round robin*".



Por último se configurará cada uno de los servidores reales encargados de procesar las solicitudes de los clientes. Únicamente es necesario indicar la IP del servidor y el puerto del servicio que se está balanceando. En este caso:

- Server1 con IP 192.168.122.21 y puerto 80 (http).
- Server2 con IP 192.168.122.22 y puerto 80 (http).





Tras haberlos configurado es necesario activarlos con la opción *"Activate"*.

Para terminar, como toda la configuración ha quedado reflejada en un archivo ubicado en el balanceador activo, es necesario copiarlo en el balanceador de respaldo, que tiene la IP 192.168.122.12, para que los dos usen la misma configuración.

```
$ scp /etc/sysconfig/ha/lvs.cf 192.168.122.12:/etc/sysconfig/ha/lvs.cf
```

9.3.2. Creación y configuración de los servidores web.

Ahora vamos a instalar el software necesario en los dos servidores web y a configurarlos. El primer paso que vamos a realizar es instalar el servidor Apache, que será el encargado de procesar las peticiones de los usuarios.

```
$ yum install httpd
$ chkconfig httpd on
$ service httpd start
```

Es necesario abrir el puerto 80 para permitir el tráfico web.

```
$ iptables -I INPUT -m state --state NEW -p tcp  --dport 80 -j ACCEPT
```

Luego instalaremos el paquete arptables_jf.

```
$ yum install -y arptables_jf
$ chkconfig arptables_jf on
$ service arptables_jf start
```

Una vez iniciado el servicio, vamos a crear las reglas necesarias que harán que los servidores reales ignoren todas las solicitudes ARP para las dirección IP virtual, y que cualquier respuesta ARP saliente que contenga la IP virtual salga con la verdadera IP del servidor. El único nodo que debe responder a las solicitudes ARP para la IP virtual es el balanceador activo.

Para TFG_WebServer1:

```
$ arptables -A IN -d 192.168.122.50 -j DROP
$ arptables -A OUT -s 192.168.122.50 -j mangle --mangle-ip-s 192.168.122.21
```

Para TFG_WebServer2:

```
$ arptables -A IN -d 192.168.122.50 -j DROP
$ arptables -A OUT -s 192.168.122.50 -j mangle --mangle-ip-s 192.168.122.21
```

Y guardamos los cambios realizados.

```
$ service arptables_jf save
```

Por último sólo queda configurar la IP virtual en los dos nodos usando para ello un alias.

```
$ ip addr add 192.168.122.50 dev eth0:1
```

Como los alias no se pueden configurar de forma permanente, es necesario crearlos cada vez que se encienden las máquinas usando el archivo /etc/rc.local.

```
$ vi/etc/rc.local
```

Y añadimos la línea:

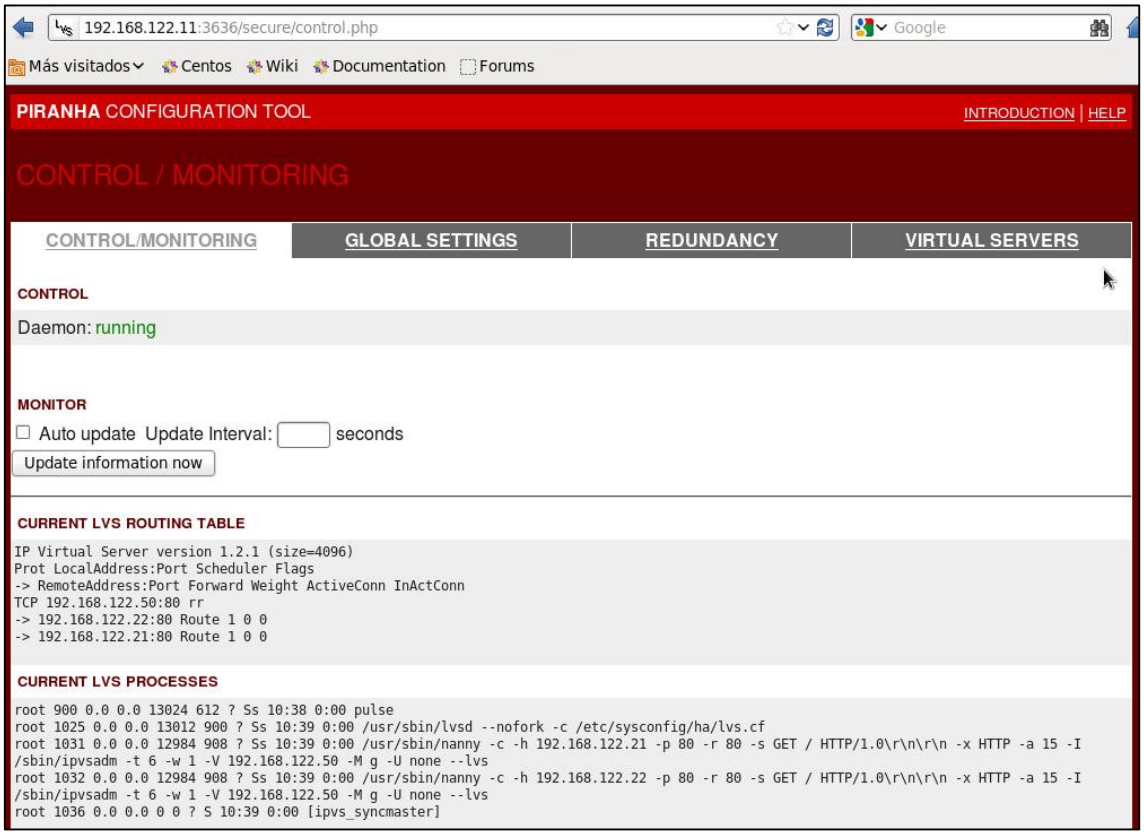
```
ip addr add 192.168.122.50 dev eth0:1
```

Se supone que con la configuración de las *"arptables"* debería haber sido suficiente, pero durante las pruebas se han detectado errores de funcionamiento, había ocasiones en las que el servicio no se balanceaba. Para evitar el problema se encontró la solución de desactivar ARP en los servidores reales. Sólo hay que acceder al archivo /etc/sysctl.conf y añadir las siguientes líneas:

```
net.ipv4.conf.all.arp_announce = 2

net.ipv4.conf.all.arp_announce = 1
```

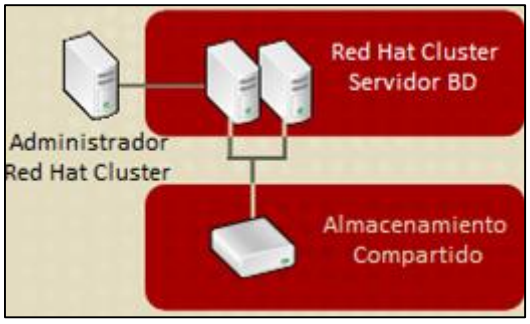
Una vez realizado ese paso es necesario hacer que los cambios tengan efecto con la orden:

```
$ sysctl -p /etc/sysctl.conf
```



9.4. Creación y configuración del clúster de base de datos en alta disponibilidad.

En este capítulo se van a describir las tareas que hay que realizar para obtener un clúster de servidores de base de datos en alta disponibilidad. Además se creará el espacio de almacenamiento compartido y distribuido que albergará la base de datos que usarán ambos servidores.



Las máquinas con las que trabajaremos son:

- TFG_AdminClusterBD: Administrador del clúster (no forma parte de él).
- TFG_ServerBD1: Servidor de base de datos activo.
- TFG_ServerBD2: Servidor de base de datos de respaldo.

9.4.1. Instalación y configuración de los módulos software en los nodos del clúster.

En todos los nodos.

Todos los nodos deben estar sincronizados. Para realizar tal tarea se usa el protocolo ntp.

```
$ yum install ntp

$ chkconfig ntpd on

$ service ntpd start
```

Nodo TFG_AdminClusterBD.

En este nodo se instala el soporte de administración del clúster.

```
$ yum groupinstall "High Availability Management"
```

Y se configura el servidor luci para que arranque automáticamente cada vez que se encienda la máquina.

```
$ chkconfig luci on
```

Finalmente se inicia el servidor luci.

```
$ service luci start
```

Nodos TFG_ServerBD1 y TFG_ServerBD2

En estos nodos se instala el software necesario para la creación del clúster.

```
$ yum groupinstall "High Availability" "Resilient Storage"
```

Se configuran los servicios correspondientes para que arranquen cada vez que se enciendan las máquinas.

```
$ chkconfig ricci on
$ chkconfig cman on
$ chkconfig clvmd on
$ chkconfig rgmanager on
$ chkconfig modclusterd on
```

Y se crea una contraseña para el usuario ricci que será usada en la creación del clúster. Por esa razón es aconsejable que en ambos nodos la contraseña de este usuario sea la misma.

```
$ passwd ricci
```

Por último se levanta el servicio.

```
$ service ricci start
```

También es necesario instalar el servidor y el cliente de la base de datos. Para ello:

```
$ yum install mysql mysql-server
```

Se habilita el servicio.

```
$ chkconfig mysqld on
```

Y lo iniciamos:

```
$ service mysqld start
```

Es una buena práctica, cuando se instala mysql, añadir una contraseña al usuario root, ya que éste viene por defecto sin ella.

```
$ mysqladmin -u root password NEWPASSWORD
```

9.4.2. Configuración del cortafuegos.

Antes de empezar con la creación del clúster debemos abrir los puertos necesarios para que los servicios se puedan comunicar.

Nodo TFG_AdminClusterBD.

Para **luci** (Servidor del usuario Conga):

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 10.22.122.0/24 -d 10.22.122.0/24 -
-dport 8084 -j ACCEPT
```

Nodos TFG_ServerBD1 y TFG_ServerBD2.

- Para **cman** (Gestor de clúster):

```
$ iptables -I INPUT -m state --state NEW -m multiport -p udp -s 10.22.122.0/24 -d
10.22.122.0/24 --dports 5404,5405 -j ACCEPT

$ iptables -I INPUT -m addrtype --dst-type MULTICAST -m state --state NEW -m
multiport -p udp -s 10.22.122.0/24 --dports 5404,5405 -j ACCEPT
```

- Para **dlm** (Gestor de bloqueo distribuido):

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 10.22.122.0/24 -d 10.22.122.0/24 -
-dport 21064 -j ACCEPT
```

- Para **ricci** (parte del agente remoto Conga):

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 10.22.122.0/24 -d 10.22.122.0/24 -
-dport 11111 -j ACCEPT
```

- Para **modclusterd** (parte del agente remoto de Conga):

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 10.22.122.0/24 -d 10.22.122.0/24 -
-dport 16851 -j ACCEPT
```

- Para **igmp** (Protocolo de administración de grupos en Internet):

```
$ iptables -I INPUT -p igmp -j ACCEPT
```

- Para **mysqld** (Servidor de base de datos):

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 192.168.122.0/24 -d
192.168.122.0/24 --dport 3306 -j ACCEPT
```

Después de ejecutar las órdenes anteriores, se guarda la nueva configuración del cortafuegos:

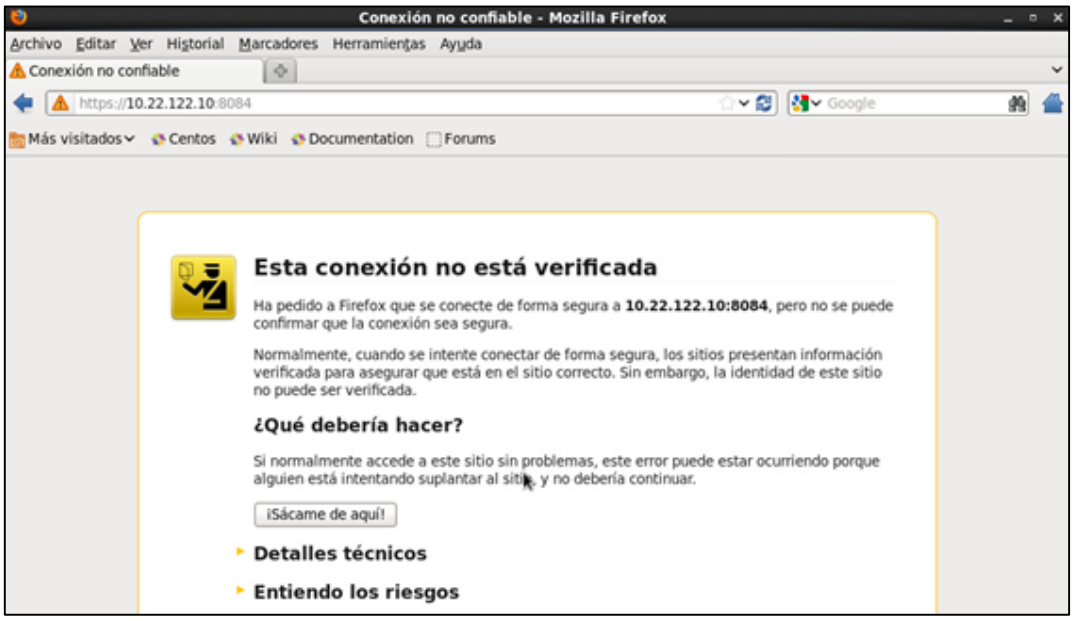
```
$ service iptables save

$ service iptables restart
```

9.4.3. Creación del clúster usando Conga.

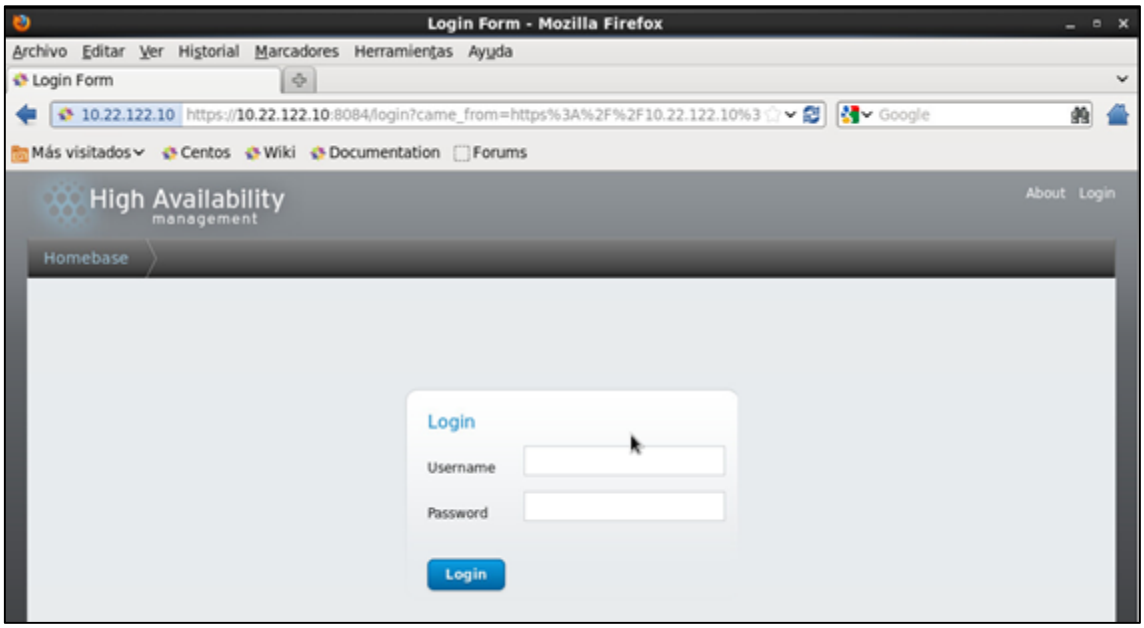
Llegados a este punto ya tenemos todos los elementos necesarios para poder empezar con la creación del clúster. Lo primero que haremos será acceder a la interfaz gráfica del Conga (servidor luci). Para ello abrimos el explorador del equipo e introducimos:

https://10.22.122.10:8084

Al acceder por primera vez nos saldrá la siguiente pantalla:



Tras haberle dado a “entiendo los riesgos” podremos “añadir la excepción” y finalmente, después de haber confirmado, aparece la interfaz gráfica en la que debemos identificarnos con nuestro nombre de usuario y contraseña.



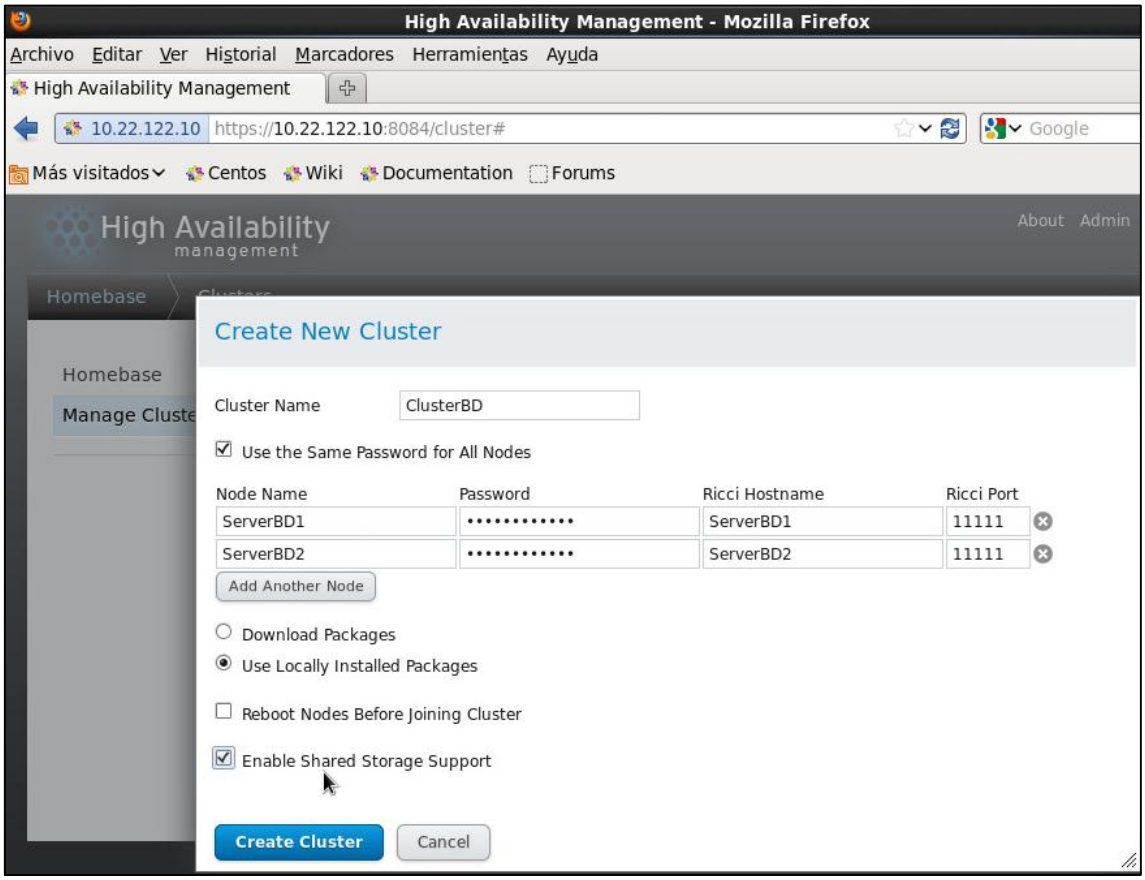
Una vez dentro nos dirigimos a “Manage Clusters” y en el apartado “create” se encuentra la pantalla para la creación del clúster.

- Ponemos como nombre del clúster “ClusterBD”.
- Marcamos la opción “Use same password for all nodes” ya que a la hora de crear los usuario ricci le pusimos la misma contraseña en todos los nodos.
- Agregamos cada una de las máquinas que conforman el clúster.
  Para facilitar el trabajo y no tener que trabajar directamente con las IPs de las máquinas, lo que se hizo fue añadir el nombre de ambas máquinas en el fichero /etc/hosts tanto el servidor luci como en los nodos que conforman el clúster. El archivo queda de la siguiente forma:

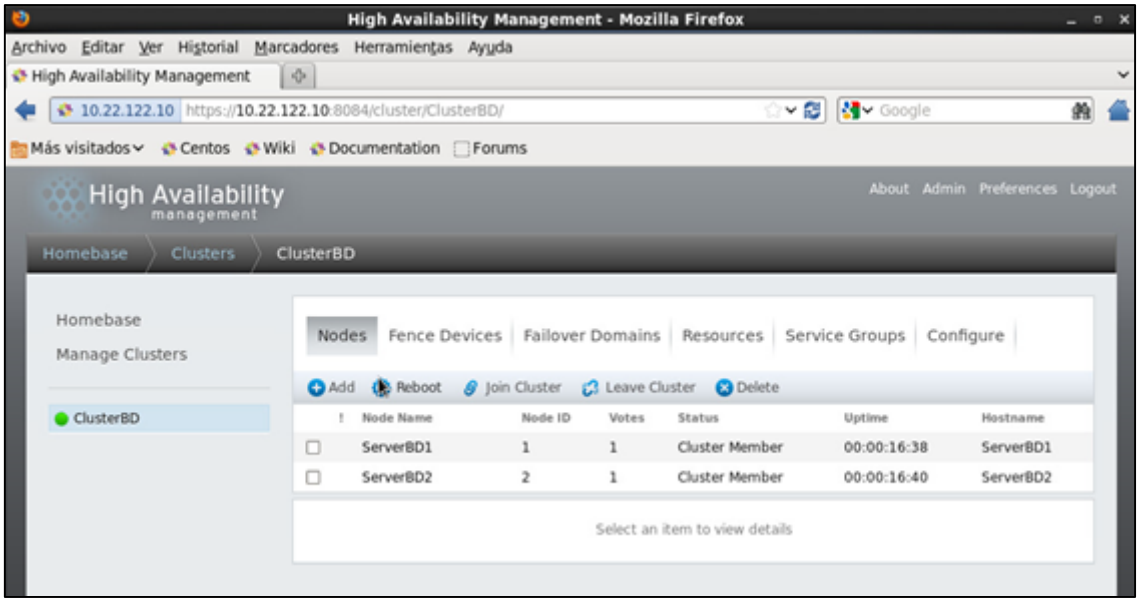
```
10.22.122.11 ServerBD1
10.22.122.12 ServerBD2
```

  Para añadir las máquinas, a parte de sus nombres es necesario la contraseña del usuario ricci y el puerto en el que escucha el servicio ricci (11111).

- Seleccionamos la opción “Use Locally Installed Packages” para usar los paquetes que nosotros mismos hemos instalado.
- Seleccionamos la opción “Enable shared storage support” para permitir el acceso a sistemas de archivos compartidos.



Finalmente, si todo ha ido bien, tendremos nuestro clúster funcionando.



9.4.4. Creación de un volumen lógico compartido.

En este apartado se van a detallar los pasos que hay que seguir para crear el espacio de almacenamiento compartido que va a utilizar el clúster. Todos estos pasos sólo habrá que hacerlos en uno de los nodos que conforman el clúster.

9.4.4.1. Creación de un volumen físico.

Lo primero que vamos a hacer es crear una partición, en el disco exportado con iSCSI (sda), con la que luego crear un volumen físico.

```
$ fdisk /dev/sda
```

Desde la línea de comandos de fdisk se crea una partición con la opción n.

```
Orden (m para obtener ayuda): n

Acción de la orden

e   Partición extendida

   p   Partición primaria (1-4)

p

Número de partición (1-4): 1

Primer cilindro (1-8205, valor predeterminado 1):

Se está utilizando el valor predeterminado 1

Last cilindro, +cilindros or +size{K,M,G} (1-8205, valor predeterminado 8205):

Se está utilizando el valor predeterminado 8205
```

Y con la opción p comprobamos que la partición se ha creado correctamente.

```
Orden (m para obtener ayuda): p

Disposit. Inicio    Comienzo      Fin      Bloques  Id  Sistema

/dev/sda1               1        8205     8401904   83  Linux
```

Finalmente con la opción w se guardan los cambios.

Luego se crea el volumen físico.

```
$ pvcreate /dev/sda1
```

- Si quisiéramos eliminar el volumen físico:
```
$ pvremove /dev/sda1
```

- Si quisiéramos ver que volúmenes físicos existen:
```
$ pvscan
```

9.4.4.2. Creación de un grupo de volúmenes.

Ahora que ya tenemos configuradas las particiones físicas, podemos crear un grupo de volúmenes que contendrá el volumen lógico**.**

Antes de hacerlo, debemos configurar el protocolo de cerrojo para que permita la creación de grupo de volúmenes y volúmenes lógicos. Para ello, en el archivo /etc/lvm/lvm.conf de los dos nodos que conforman el clúster (TFG_ServerBD1 y TFG_ServerBD2), modificamos el valor de la variable locking_type, dejándolo de la siguiente manera:

```
locking_type=0
```

Una vez echo eso, ya podemos crear el grupo de volúmenes usando:

```
$ vgcreate -c y Grupo_Almacen /dev/sda1
```

- Si quisiéramos eliminar el grupo de volúmenes:
```
$ vgremove Grupo_Almacen
```

- Si quisiéramos ver que grupo de volúmenes existe:
```
$ vgscan
```

9.4.4.3. Creación de un volumen lógico.

Llegados a este punto, vamos a crear el volumen lógico con la orden:

```
$ lvcreate -l 100%FREE -n Volumen_Almacen Grupo_Almacen
```

- Si quisiéramos eliminar el volumen lógico:
```
$ lvremove /dev/sda
```

- Si quisiéramos ver que volúmenes lógicos existen:
```
$ lvscan
```

9.4.4.4. Creación de un sistema de archivos compartido.

Ahora que ya se dispone de un volumen lógico, vamos a crear un sistema de archivos compartido en él.

```
$ mkfs.gfs2 -p lock_dlm -t ClusterBD:storage -j 2 /dev/Grupo_Almacen/Volumen_Almacen
```

Donde:

- -p lock_dlm: Especifica el protocolo de sincronización para acceder al sistema de archivos.
- -t ClusterBD:storage: ClusterBD es el nombre del clúster que se ha creado antes y que utilizará el sistema de archivos al se le asignará la etiqueta storage.
- -j 2: Número de nodos que podrán acceder al espacio compartido.
- /dev/Grupo_Almacen/Volumen_Almacen: Ruta del volumen lógico creado.

9.4.5. Montaje del sistema de archivos distribuido.

Llegados a este punto vamos a hacer que los dos servidores de base de datos monten el sistema de archivos que se ha creado anteriormente para que puedan usarlo. Para ello lo primero que vamos a hacer en ambos nodos es crear un punto de montaje.

```
$ mkdir /Almacen
```

Luego accedemos al fichero /etc/fstab para hacer que el sistema de archivos se monte cada vez que se arranque la máquina.

```
UUID=c2c08f55-a23e-4c8f-b07f-332cf8d063f8  /              ext4    defaults          1 1
UUID=750ef787-165c-4ad8-9795-189a3ae7c9a6  swap           swap    defaults          0 0
tmpfs                                      /dev/shm       tmpfs   defaults          0 0
devpts                                     /dev/pts       devpts  gid=5,mode=620    0 0
sysfs                                      /sys           sysfs   defaults          0 0
proc                                       /proc          proc    defaults          0 0
/dev/Grupo_Almacen/Volumen_Almacen         /Almacen       gfs2    defaults          0 0
```

9.4.6. Configuración del servicio Mysql en alta disponibilidad.

En este momento disponemos de un clúster básico que ya puede usar el almacenamiento compartido. Así que ahora vamos a poner el servicio en alta disponibilidad. Como vamos a hacer que determinados servicios sean manejados por el propio clúster, los vamos a parar en los dos nodos que lo conforman.

```
$ chkconfig clvmd off

$ chkconfig iscsid off

$ chkconfig gfs2 off

$ chkconfig mysqld off
```

9.4.6.1. Creación de los recursos controlados por el clúster.

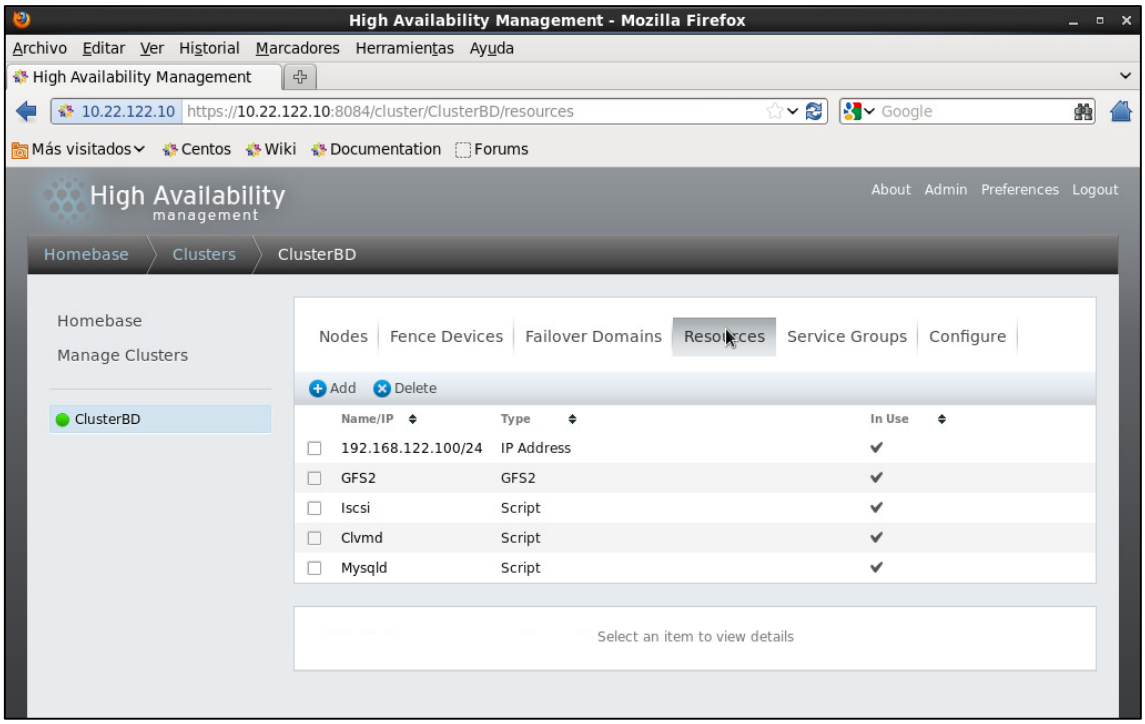
Para agregar los recursos que necesita manejar el clúster nos dirigimos a la pestaña *"Resources"* y allí los crearemos con los siguientes datos:

Recurso	Nombre	Path
Script	Iscsi	/etc/init.d/iscsid
Script	Clvmd	/etc/init.d/clvmd
Script	Httpd	/etc/init.d/httpd
Script	Mysqld	/etc/init.d/mysqld

Recurso	IP Address	Netmask
IP Address	192.168.122.100	24

Recurso	Nombre	Punto de montaje	Dispositivo	Sistema de archivos	Forzar desmontaje
GFS2	GFS2	/Almacen	/dev/Grupo_Almacen/Volumen_Almacen	GFS2	Activo

Al finalizar deberíamos tener algo parecido a esto:

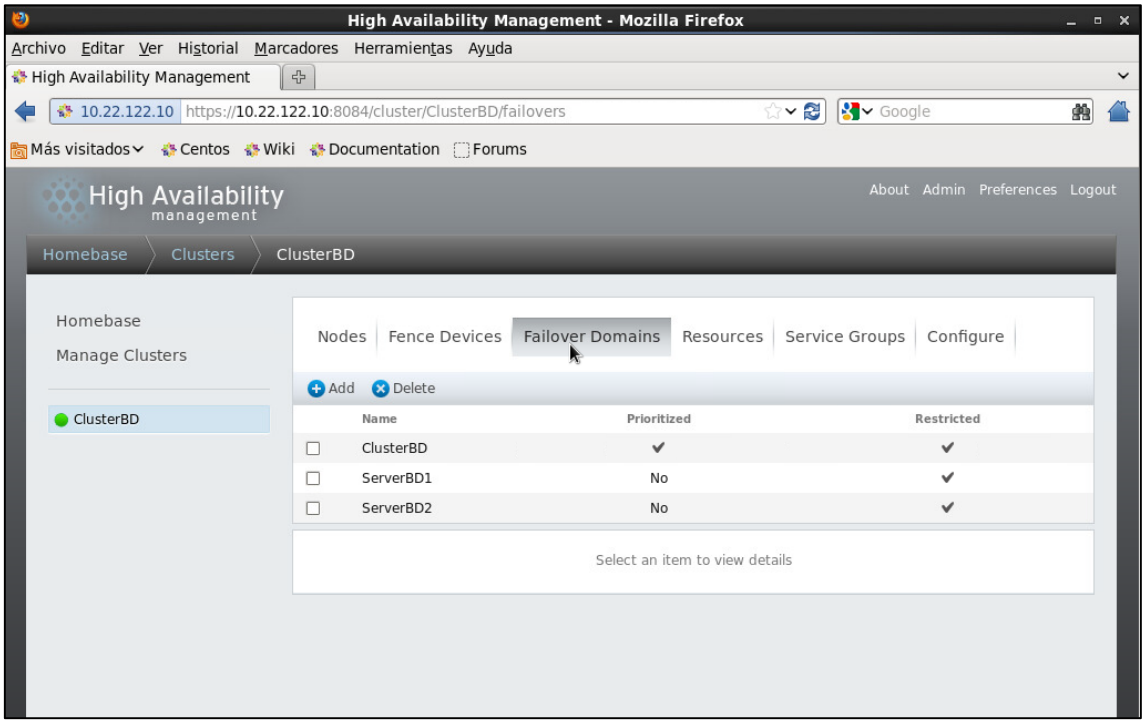


9.4.6.2. Creación de los dominios "failover".

Ahora vamos a crear los dominios *"failover"* que se encargarán de restringir en qué nodos puede ejecutarse un determinado servicio. Para realizar esto, accedemos a la pestaña *"Failover Domains"* para añadir los que se muestran a continuación:

Nombre	Con prioridad	Restringido	Sin conmutación por recuperación	Nodos	Prioridad
ClusterBD	-	Activado	-	ServerBD1	-
ServerBD1	-	Activado	-	ServerBD2	-
ServerBD2	Activado	Activado	Activado	ServerBD1	1
				ServerBD2	2

Una vez se han agregado cada uno de los *"failovers"* necesarios, tendremos algo como esto:



9.4.6.3. Creación de los grupos de servicios.

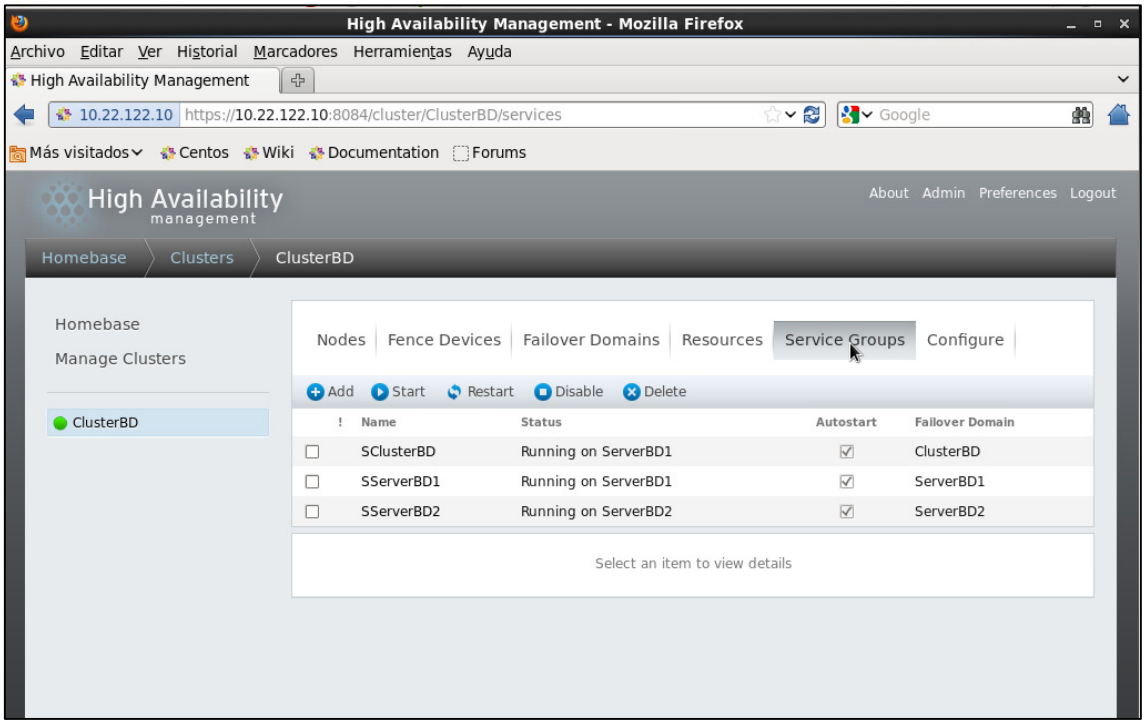
Para terminar con la creación del clúster de servidores de base de datos en alta disponibilidad, sólo nos queda crear los grupos de servicios. Esto lo haremos a través de la pestaña *"Service Groups"*.

Nombre del servicio	Arranque automático	Failover	Politica de recuperación	Recursos
SClusterBD	Activado	ClusterBD	Recolocar	IP Address
				Mysqld
SServerBD1	Activado	ServerBD1	Reiniciar	Iscsi
				Clvmd
				Gfs2
SServerBD2	Activado	ServerBD2	Reiniciar	Iscsi
				Clvmd
				Gfs2

Hay que tener en cuenta las siguientes consideraciones:

- Para añadir un recurso se usa la opción "*Add resource*".
- Es importante que la política de recuperación del servicio SClusterBD sea recolocar para que la IP y el servicio de base de datos migre al nodo de respaldo cuando el activo falle.
- Otra cosa importante es anidar los recursos Iscsi, Clvmd y GFS2 para que se inicien en este mismo orden. Para ello una vez tenemos añadido el primer recurso, usamos la opción "*Add child resource*".

El resultado final quedaría:



9.5. Instalación de la tienda virtual Prestashop.

Llegados a este punto ya estamos en condiciones de poder desplegar el servicio de tienda virtual.

9.5.1. Instalación de los módulos software necesarios en los servidores web.

Antes de nada vamos a instalar en ambos servidores web PHP y algunas de sus extensiones:

```
$ yum install php php-mysql
```

Instalación de la librería GD:

```
$ yum install php-gd gd-devel
```

Instalación de la extensión Mcrypt:

Para la instalación de esta librería ha sido necesario descargase el repositorio EPEL.

```
$ yum install wget
$ wget http://download.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm
$ rpm -ivh epel-release-6-8.noarch.rpm
$ yum install php-mcrypt
```

Instalación de la extensión Mbstring:

```
$ yum install php-mbstring
```

Instalación de la extensión Dom:

```
$ yum isntall php-xml
```

9.5.2. Creación de la base de datos y su usuario.

Como la base de datos se alojará en el espacio compartido, lo primero que tenemos que hacer es compartir el archivo de configuración de Mysql para que ambos servidores de base de datos usen el mismo. Para ello en uno de ellos:

```
$ cp /etc/my.conf /etc/my.cnf_copia
$ mv /my.conf /Almacen/my.conf
$ ln -s /Almacen/my.conf /etc/my.conf
```

Y en el otro:

```
mv /etc/my.conf /etc/my.cnf_copia
$ ln -s /Almacen/my.conf /etc/my.conf
```

Ahora que ya ambos servidores usan la misma configuración debemos hacer que la ruta de datos apunte al directorio donde queremos que se cree la base de datos compartida.

```
$ mkdir /Almacen/Prestashop

$ vi /Almacen/my.cnf
```

```
[mysqld]
datadir=/Almacen/Prestashop
socket=/var/lib/mysql/mysql.sock
# Disabling symbolic-links is recommended to prevent assorted security risks;
symbolic-links=0

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

Es muy importante que después de haber modificado el archivo de configuración se reinicie el servicio para que los cambios tengan efecto.

```
$ service mysqld restart
```

Ahora ya podemos crear la base de datos y el usuario que puede acceder a ella. Al usar el espacio compartido sólo es necesario hacerlo en uno de los nodos.

```
$ mysql -u root -p
```

```
mysql> CREATE DATABASE prestashop

mysql> CREATE USER 'prestashop'@'localhost' IDENTIFIED BY 'contraseña';
mysql> GRANT ALL PRIVILEGES ON *.* TO 'prestashop'@'localhost'    WITH GRANT OPTION;

mysql> CREATE USER 'prestashop'@'%' IDENTIFIED BY 'contraseña';
mysql> GRANT ALL PRIVILEGES ON *.* TO 'prestashop'@'%' WITH GRANT OPTION;

mysql> FLUSH PRIVILEGES;
```

9.5.3. Instalación.

El proceso de instalación se llevará a cabo con sólo un servidor web y de base de datos funcionando.

Lo primero que tenemos que hacer es descargar el software de la tienda virtual en el directorio raíz de Apache y descomprimirlo.

```
$ cd /var/www/html
```

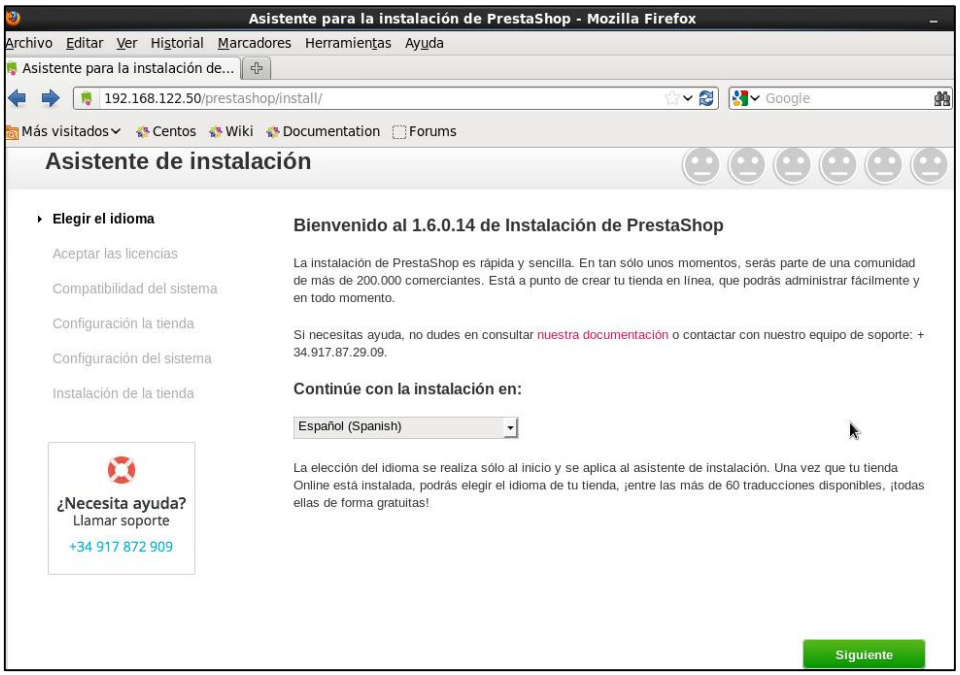
```
$ wget
https://www.prestashop.com/ajax/controller.php?method=download&type=releases&file=pre
stashop_1.6.0.14.zip&language=es
```

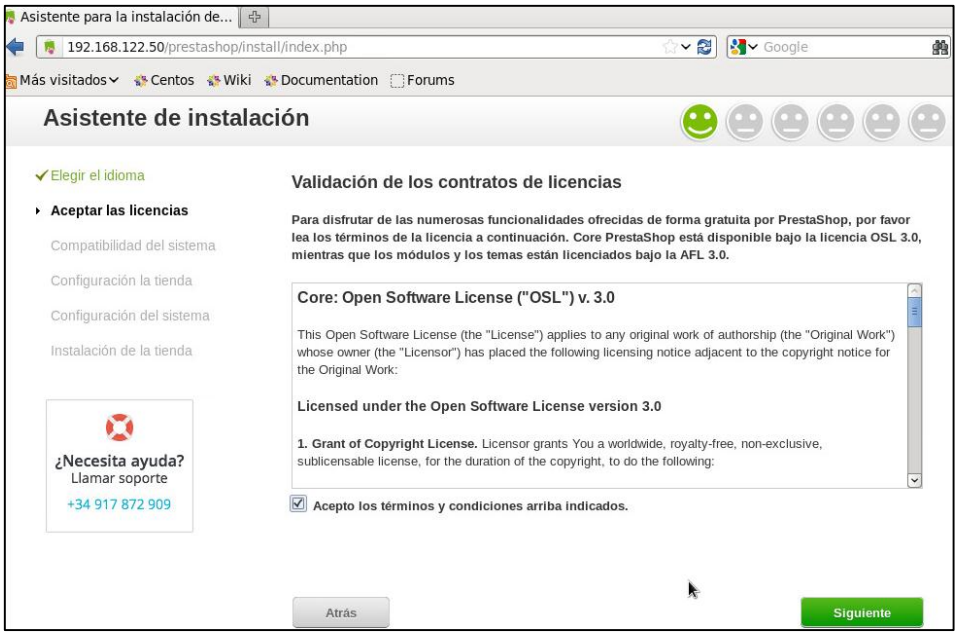
```
$ unzip prestashop_1.6.0.14.zip
```

Luego, tal y como se indica en la página de instalación de Prestashop, debemos hacer que el propietario de la carpeta sea Apache y darle los permisos, representados en notación octal, adecuados: 775 para directorios y 644 para archivos regulares.

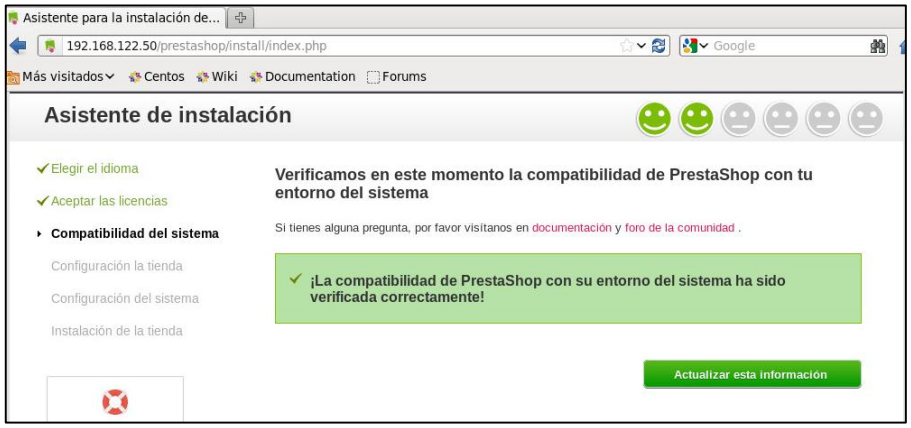
```
$ chown -R apache:apache /var/www/html/

$ find /var/www/html/ -type d -exec chmod 775 {} \;

$ find /var/www/html/ ! -type d -exec chmod 664 {} \;
```

Una vez tengamos eso hecho, usando el navegador web del sistema anfitrión, accedemos al asistente para comenzar la instalación. Lo primero que haremos será seleccionar el idioma y aceptar las licencias.

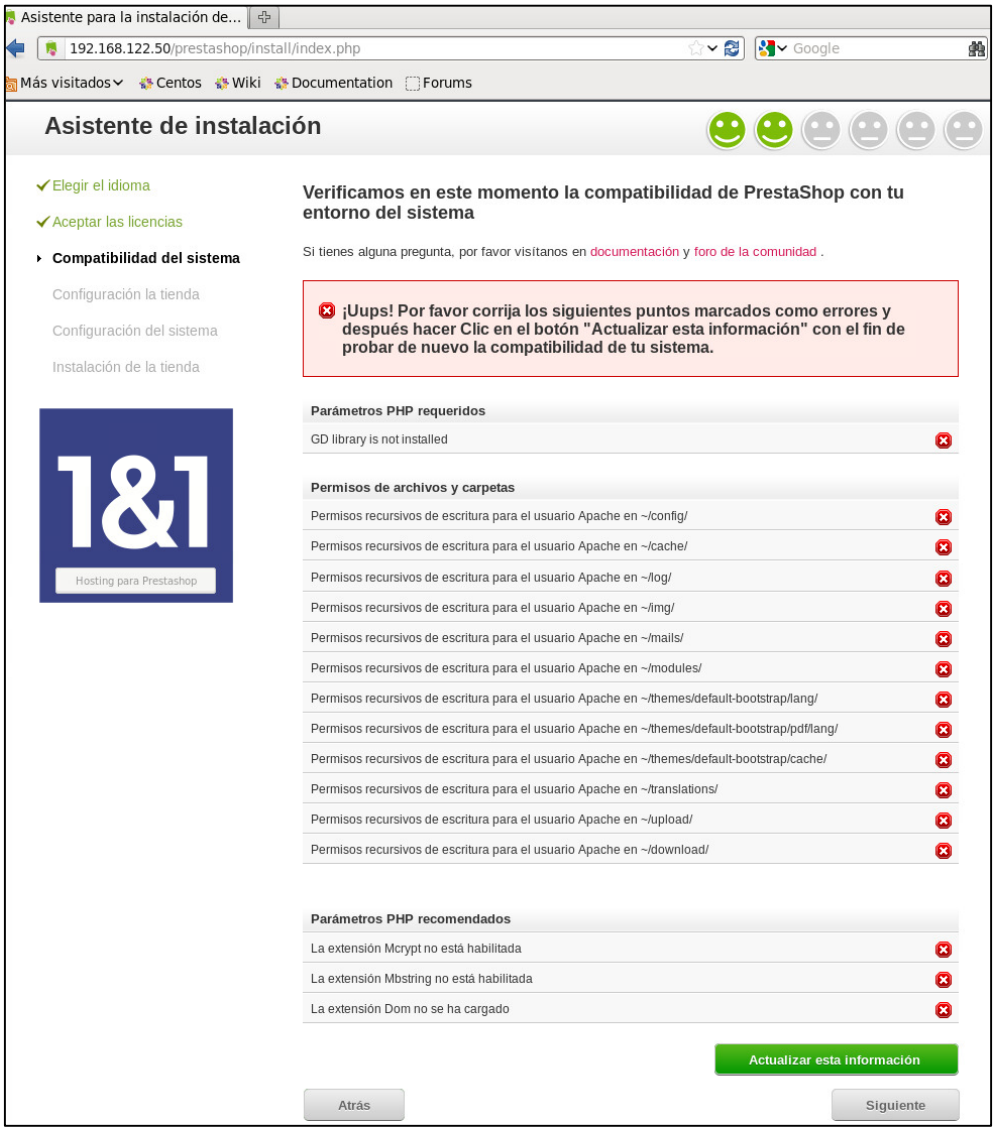




En este momento si todo se ha realizado correctamente, la compatibilidad de Prestashop con su entorno debe haber sido verificada correctamente.

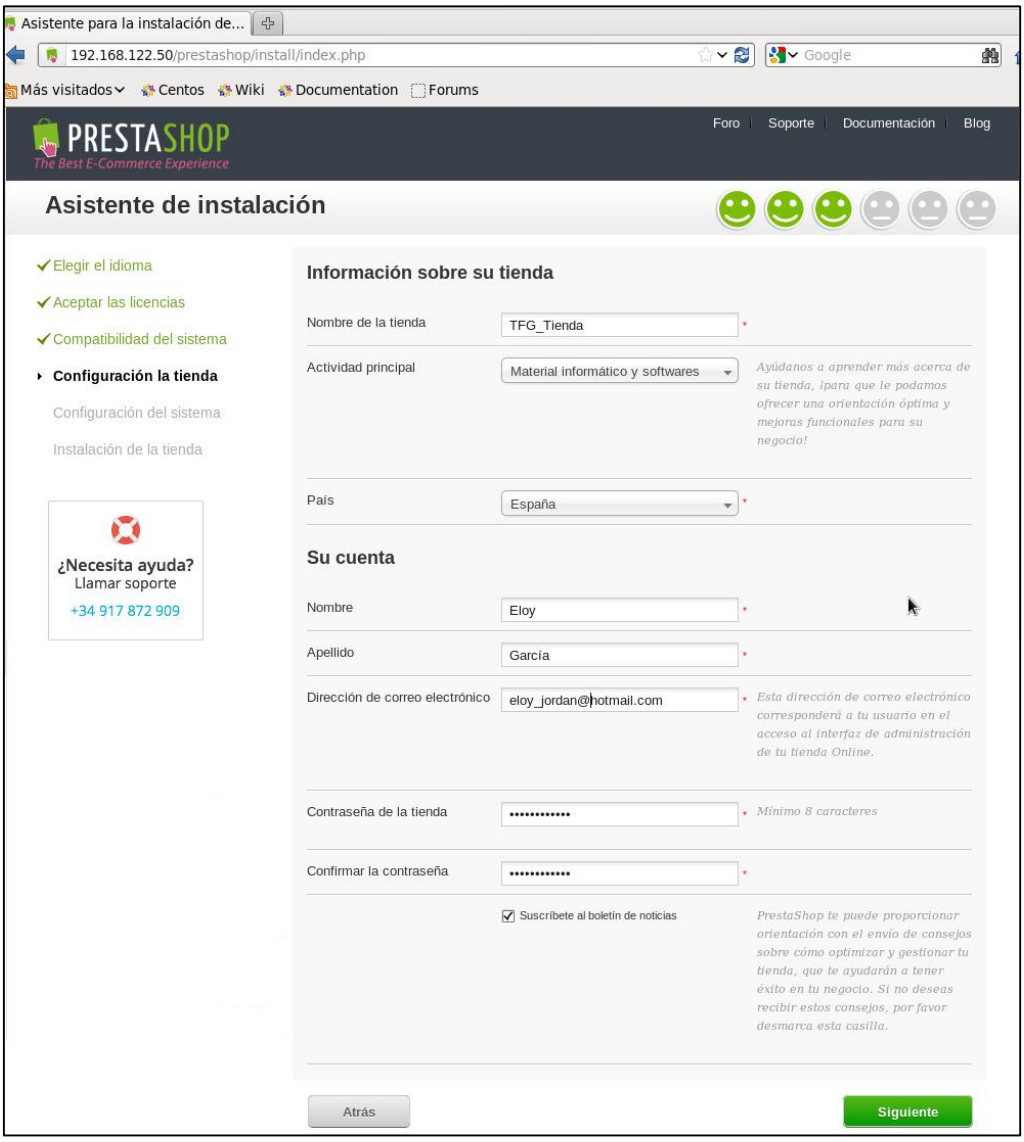


En caso contrario se mostraría una pantalla indicándonos el error cometido.



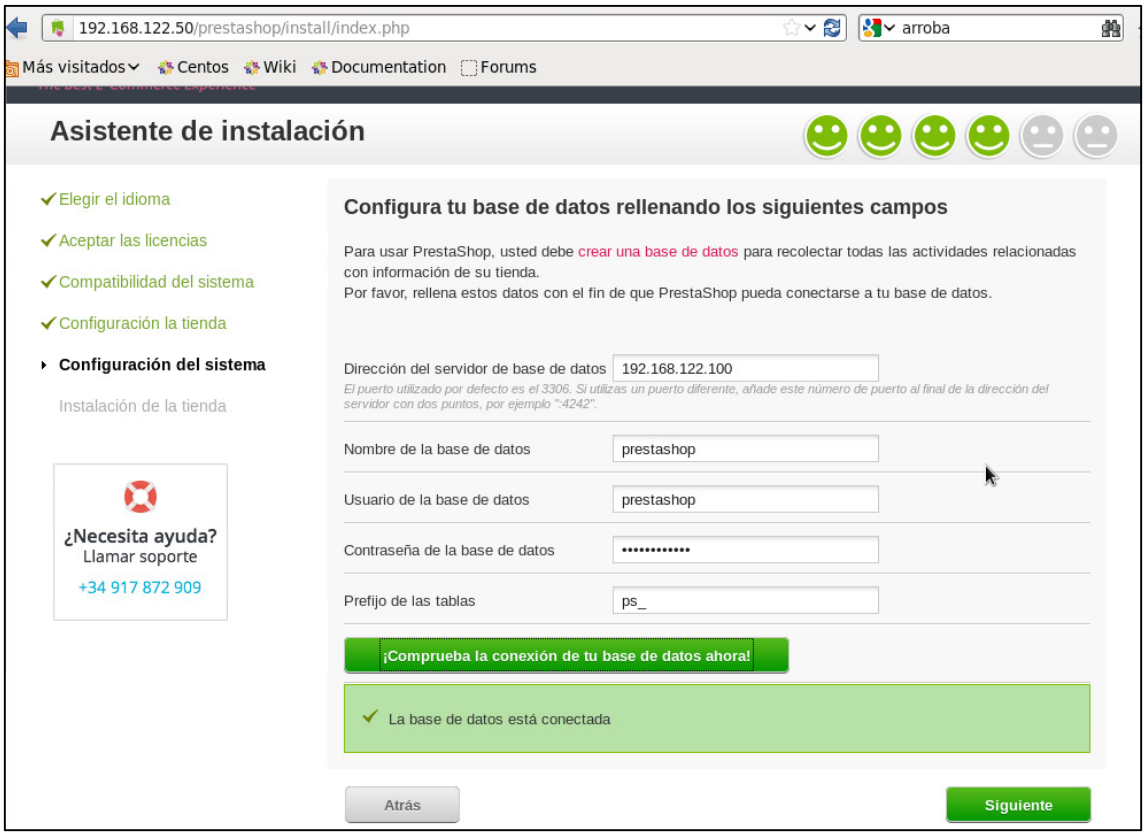
A continuación tendremos que darle un nombre a la tienda e indicar el país al que pertenece. Además también habrá que crear la cuenta del administrador con los siguientes datos:

- Nombre.
- Apellido.
- Correo.
- Contraseña.

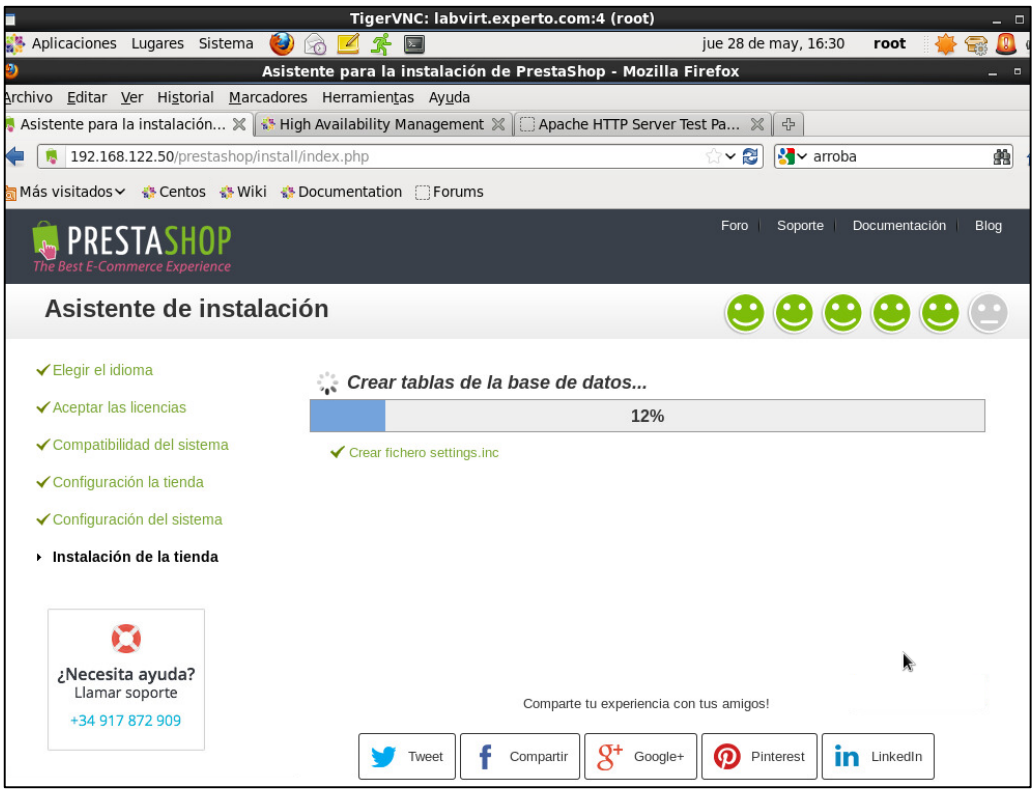


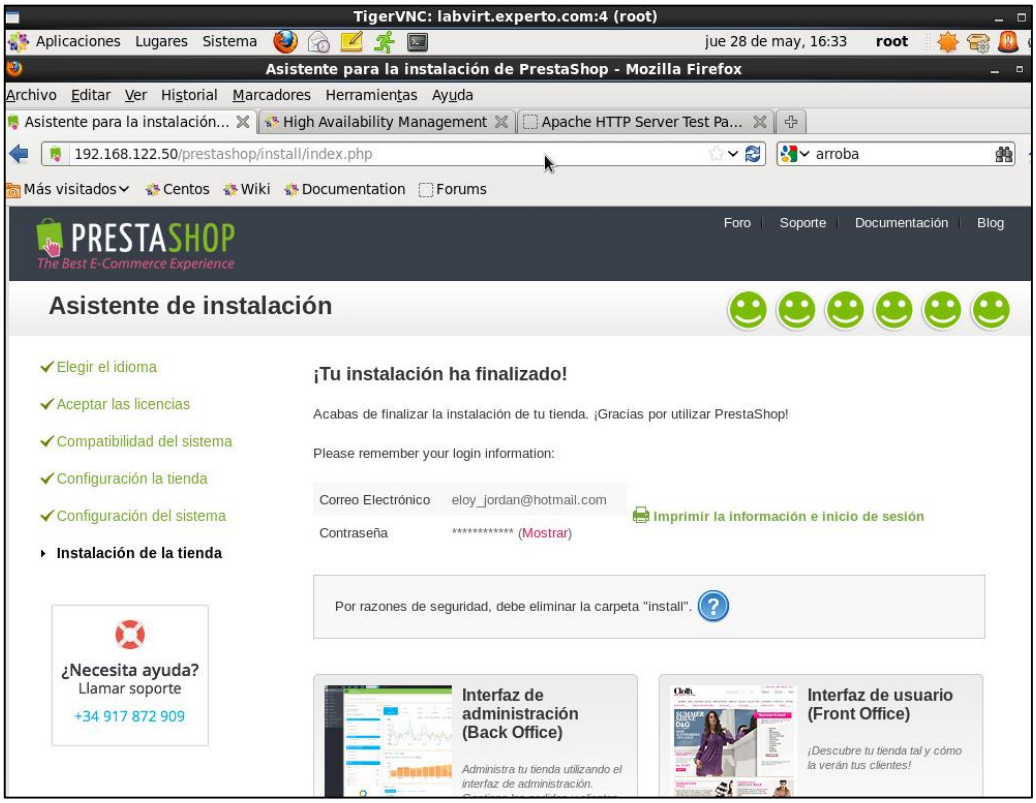
Por último habrá que configurar la conexión con la base de datos. Para hacerlo se nos solicitará:

- Dirección del servidor.
- Nombre de la base de datos.
- Nombre del usuario.
- Contraseña.



Ahora solo toca esperar a que se realice la instalación de Prestashop.





Una vez se ha finalizado la instalación por razones de seguridad se recomienda borrar la carpeta install.

```
$ rm -Rf /var/www/html/prestashop/install
```

Como la instalación sólo se ha realizado en un nodo, tenemos que copiar la carpeta resultante al otro servidor web. Una cosa importante que hay que hacer antes de realizar la copia es acceder como administrador ya que esta acción modifica el nombre de la carptea admin. Para ello desde el navegador web entramos en http://192.168.122.50/prestashop/admin. Con el fin de evitar que se modifiquen los permisos o el propietario de la carpeta prestashop, la comprimimos y luego la copiamos al otro servidor.

En el nodo dónde se ha realizado la instalación (TFG_WebServer1):

```
$ cd /var/www/html
$ tar czvf prestashop.tar.gz prestashop
$ scp prestashop 192.168.122.22:/var/www/html
```

En el nodo TFG_WebServer2 sólo tenemos que descomprimir la carpeta que acabamos de pasar:

```
tar xzvf prestashop.tar.gz
```

10. Pruebas.

Errar es humano y la etapa de pruebas tiene como objetivo detectar los errores que se hayan podido cometer a lo largo de todo el desarrollo del sistema de información. Es de vital importancia corregirlos para así cumplir con lo que el usuario espera del sistema desde el punto de vista de su funcionalidad y de su rendimiento.

10.1. Pruebas de unidad.

Las pruebas de unidad sirven para comprobar el correcto funcionamiento de cada componente que conforma nuestro sistema.

10.1.1. Pruebas realizadas al clúster de servidores web con balanceo de carga.

A continuación se detallarán las pruebas que se han realizado para asegurar el adecuado funcionamiento de este subsistema.

10.1.1.1. Comprobación de la infraestructura de red.

Tras haber configurado las interfaces de red y el cortafuegos de todos los nodos se comprueba que éstos tienen acceso entre ellos.

En cada nodo se ejecuta:

```
$ ping -I eth0 192.168.122.11

$ ping -I eth0 192.168.122.12

$ ping -I eth0 192.168.122.21

$ ping -I eth0 192.168.122.22
```

También es necesario, después de haber creado los LVS, probar el correcto funcionamiento de la IP virtual. Esto se puede hacer realizando un ping a dicha dirección o bien accediendo al servidor web a través de un navegador.

```
$ ping -I eth0 192.168.122.50
```

Para terminar con este apartado sólo nos queda comprobar que los alias en las interfaces de red de los servidores reales han sido creados correctamente y por lo tanto que en el balanceador activo existen las rutas hacia ellos. Para verlo podemos usar:

```
$ ipvsadm
```

```
$ watch ipvsadm
```

La diferencia entre ambas órdenes es que la primera te muestra los resultados en un momento determinado mientras la segunda es en tiempo real.

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.122.50:http rr
  -> 192.168.122.21:http          Route   1      0          2
  -> 192.168.122.22:http          Route   1      0          3
```

10.1.1.2. *Comprobación del correcto funcionamiento del balanceo de carga.*

Después de haber configurado los LVS y haber creado los alias de las interfaces de los servidores web reales, se está en condiciones de ver si el balanceo funciona correctamente. Con este fin se crean dos páginas de prueba que se colocan en cada uno de los servidores web en el directorio /var/www/html. Ahora si accedemos a ellos a través de dos navegadores distintos vemos como uno de ellos muestra una página y el otro muestra otra distinta. También se podría haber pulsado repetidamente f5 en uno de los navegadores y ver como se alternan las dos páginas distintas.

Otra forma de comprobar si realmente las peticiones se están repartiendo entre los dos servidores es a través de la orden:

```
$ ipvsadm
```

Si nos fijamos en las conexiones vemos como el balanceador las ha repartido entre los dos servidores.

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.122.50:http rr
  -> 192.168.122.21:http          Route   1      126        13986
  -> 192.168.122.22:http          Route   1      163        13948
```

Por último también se podría haber recurrido a los ficheros de log de apache:

```
$ cat /etc/httpd/logs/access_log | wc -l
```

Si partimos de unos ficheros de log vacíos, el número de accesos registrados en cada servidor web real debería ser la mitad del total de accesos al LVS.

10.1.1.3. Comprobación de la recuperación ante fallos.

Para ver cómo se comporta nuestro subsistema ante la caída de alguno de los nodos, se van a estudiar los siguientes casos:

- Caída del balanceador activo (TFG_LoadBalancer1): Las peticiones se siguen repartiendo entre los dos servidores web ya que el balanceador de respaldo (TFG_LoadBalancer2) toma el control.
- Recuperación del servidor caído: Si se recupera el balanceador que anteriormente se había caído, éste se convierte en el de respaldo.
- Caída del servidor web (TFG_WebServer1): Si se cae el servidor web, el balanceador lo detecta y empieza a mandar las peticiones al único servidor activo (TFG_WebServer2). El rendimiento decae.
- Recuperación del servidor web (TFG_WebServer1): Si se recupera el servidor caído, el balanceador lo detecta y vuelve a repartir las solicitudes entre los dos nodos.

Las pruebas anteriores se repiten haciendo que se caigan y recuperen las máquinas que antes no lo habían hecho. Los resultados son idénticos a los vistos antes.

10.1.1.4. Comprobación del rendimiento.

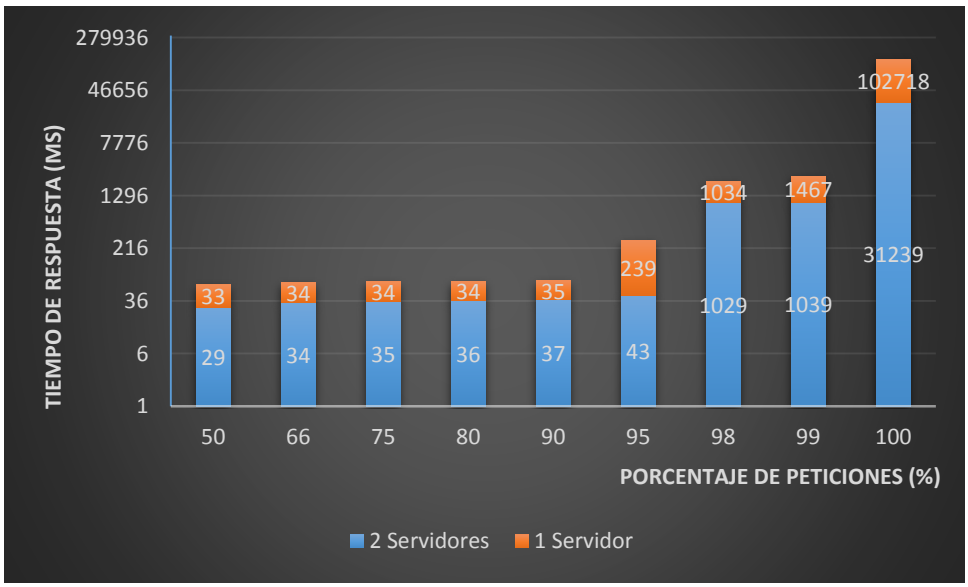
Usando la aplicación Apache Bench se mide el rendimiento del clúster de la siguiente forma:

```
ab -n 500000 -c 500 -r http://192.168.122.50/
```

Donde:

- -n 500000: indica el número de peticiones realizadas.
- -c 500: indica el nivel de concurrencia.
- -r: con esta opción no se cierra el socket si se reciben errores.
- http://192.168.122.50/ es la URL donde se ubica el servidor web.

Luego se realiza la misma prueba con sólo un servidor activo. Estos son los resultados de ambas pruebas:



	2 Servidores web	1 Servidor web
Tiempo del test (s)	67,063	127,401
Respuestas completas	500000	500000
Respuestas fallidas	0	643
Bytes totales transferidos	177000000	176924244
HTML transferido (bytes)	43500000	43481382
Respuestas por segundo	7455,71	3924,63
Tiempo por respuesta (ms)	0,134	0,255
Velocidad de transferencia	2577,46	13656,18

Como era de esperar el rendimiento con los dos servidores funcionando es prácticamente el doble que con uno sólo.

10.1.2. Pruebas realizadas al clúster de servidores de base de datos en alta disponibilidad.

Ahora vamos a ver las pruebas que se han realizado al otro subsistema que conforma nuestro sistema de información, el clúster de bases de datos en alta disponibilidad.

10.1.2.1. Comprobación de la infraestructura de red.

Ya que se ha creado una red aislada para comunicar a los nodos que conforman el clúster, vamos a comprobar que éstos sólo pueden hacerlo a través de esta red. En la máquina TFG_ServerBD1 sería:

```
$ ping -I eth0 10.22.122.12
$ ping -I eth1 10.22.122.12
$ ping -I eth2 10.22.122.12
```

Sólo se obtiene la respuesta de la interfaz eth1 que es la que corresponde con la red aislada.

Además será necesario comprobar que los dos nodos tienen acceso a la cabina de discos. Al igual que antes se usa la orden ping debiendo obtener una respuesta.

En los dos nodos:

```
$ ping -I eth2 10.22.146.190
```

10.1.2.2. *Comprobación del almacenamiento distribuido.*

Durante la creación del almacenamiento distribuido se han ido realizando las siguientes pruebas:

- Se comprueba que se ha exportado correctamente el disco.

```
$ tail /var/log/messages
```

- Se comprueba que se ha creado correctamente el volumen físico.

```
$ pvscan
```

- Se comprueba que se ha creado correctamente el grupo de volúmenes.

```
$ vgscan
```

- Se comprueba que se ha creado correctamente el volumen lógico.

```
$ lvscan
```

- Después de formatear el volumen lógico y montarlo en los dos nodos se comprueba si la información que se añade al disco en uno de los dos nodos aparece también en el otro y que esta información es persistente.

10.1.2.3. *Comprobación de la recuperación ante fallos.*

Ahora vamos a probar si el clúster de alta disponibilidad realmente funciona. Lo primero que se hace es comprobar si es capaz de activar los servicios. Luego se comprueba que si se cae el servidor de base de datos activo el servicio migra al de respaldo. En ambos casos el resultado ha sido el correcto.

10.2. Pruebas de integración.

Las pruebas de integración son las que se realizan cuando vamos juntando los componentes que conforman nuestro sistema y cuyo funcionamiento ha sido probado.

A la hora de unir los dos clúster lo único que hay que tener en cuenta es que los servidores web puedan acceder a los servidores de base de datos a través de la IP virtual de estos últimos. Para ello, usando la orden ping en cada uno de los servidores web, se comprueba que los servidores de base de datos responden.

```
$ ping -I eth0 192.168.122.100
```

Como el clúster es el encargado de manejar los servicios, para realizar la comprobación con el servidor de respaldo, habrá que forzar que el servicio migre a éste, siendo necesario para ello apagar el servidor activo.

10.3. Pruebas del sistema.

Las pruebas de sistema tienen por objetivo comprobar que el sistema, que ha superado las pruebas de integración, se comporta correctamente con su entorno.

Una vez se ha instalado Prestashop estamos en condiciones de probar el sistema como un todo. Para ello se realizan muchas de las pruebas que se habían hecho a cada subsistema por separado y se comprueba que:

- El rendimiento del sistema de información final es prácticamente el mismo al que habíamos visto antes. Es un poco menor puesto que antes se mostraba la página de inicio de Apache y ahora se muestra la de Prestashop que es un poco más pesada.

```
ab -n 500000 -c 500 -r http://192.168.122.50/prestashop/
```

- Si se cae un balanceador activo toma el control el de respaldo.
- Si se cae uno de los servidores web el otro sigue proporcionando el servicio.
- Si se cae el servidor de base de datos activo toma el control el de respaldo.
- Si cualquiera de los servidores caídos vuelve a ponerse en marcha el sistema sigue funcionando correctamente.
- Para que el servicio deje de estar disponible tiene que darse una de las siguientes opciones:
    - Los dos servidores de base de datos tienen que caerse.
    - Los dos servidores web tienen que caerse.
    - Los dos balanceadores tienen que caerse.

Para terminar se ha entrado tanto en la parte pública como en la privada de la tienda virtual y se ha navegado por todas las opciones que ésta ofrece, no habiendo detectado ningún tipo de error.

11. Conclusiones y trabajos futuros.

11.1. Conclusiones.

El hecho de construir un sistema de información como este no es una tarea fácil. Son muchas las consideraciones a tener en cuenta y conseguir que todo el conjunto funcione correctamente puede resultar un tanto complejo. Han sido muchas las horas invertidas en su realización. Gracias a todo el esfuerzo invertido se ha logrado cumplir con los objetivos marcados y en el tiempo establecido.

Enfrentarse a un proyecto de estas características ha supuesto un reto tanto personal como profesional. Ser uno mismo el encargado de planificar, analizar, diseñar, construir y probar una solución que se ajustara a la descripción del proyecto ha sido una experiencia nueva y muy gratificante.

Este proyecto me ha servido para mejorar mi formación, gracias a él he podido ampliar mi conocimiento sobre las tecnologías de virtualización y de alta disponibilidad. Además me ha permitido experimentar con otro tipo de tecnologías que hasta ahora no me resultaban muy conocidas, como fue el hecho de usar balanceo de carga y el tener que trabajar con una cabina de discos.

Por todo ello estoy muy satisfecho con el trabajo realizado y con el resultado obtenido. Se ha diseñado, instalado, configurado y desplegado un servicio de tienda virtual que funciona en alta disponibilidad y con calidad de servicio en lo que se refiere a sus tiempos de respuesta. Para lograrlo se ha tenido que:

- Diseñar la topología del clúster y de las redes de datos utilizadas.
- Crear, instalar y configurar los elementos de infraestructura necesarios: redes de datos, máquinas virtuales y cabinas de almacenamiento y productos software para proporcionar alta disponibilidad y rendimiento en computación y almacenamiento.

11.2. Trabajo futuro.

Tras haber terminado con el proyecto y haber obtenido el resultado deseado, quedan varios aspectos de éste que podrían mejorarse:

- Se ha trabajado con un diseño de red muy simple ya que no se había marcado como un objetivo. Sería importante realizar un diseño más elaborado de la infraestructura de red y trabajar con más redes aisladas de tal forma que sólo permitiera las comunicaciones entre las máquinas que lo necesitan.

- Desde la página de Red Hat Cluster Suite nos aconsejan deshabilitar SELINUX ya que si se usa con GFS2 reduce su velocidad. Habría que ver en qué medida cae la velocidad para poder valorar si configurar o no SELINUX.

- Se debería obtener un certificado SSL y configurar apache para poder usar el protocolo HTTPS para que las conexiones fuesen seguras.

- Habría que asignarle un nombre de dominio a la IP que ofrece el servicio en el DNS de la universidad y hacerlo accesible desde cualquier punto de la universidad.

- El volumen que se ha exportado de la cabina está sin contraseña, sería bueno añadirle una y configurar iSCSI para que pueda seguir accediendo a él. Además estaría bien crear un nuevo volumen en la cabina para comprobar cómo se añade un nuevo disco al grupo de volúmenes y se expande el volumen lógico.

12. Fuentes de información.

- [1] Virtualización:
http://www.adminso.es/images/6/6d/Eugenio_cap1.pdf

- [2] Sistemas de computación distribuidos:
http://searchdatacenter.techtarget.com/es/definicion/Computacion-en-la-nube
http://es.wikipedia.org/wiki/Computaci%C3%B3n_distribuida

- [3] Tipos de clúster:
https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/5/html/Cluster_Suite_Overview/ch.gfscs.cluster-overview-CSO.html

- [4] Sistemas de almacenamiento:
http://www.byspel.com/sistemas-de-almacenamiento/

- [5] Normativa y legislación:
http://www.agpd.es/portalwebAGPD/canaldocumentacion/legislacion/union_europea/directivas/index-ides-idphp.php
http://es.wikipedia.org/wiki/ISO/IEC_27000-series
http://hipertextual.com/archivo/2014/05/como-elegir-licencias-open-source/

- [6] Libvirt:
https://docs.fedoraproject.org/es-ES/Fedora/18/html/Virtualization_Getting_Started_Guide/sec_libvirt-libvirt-tools.html
https://libvirt.org/index.html
http://linux.die.net/man/1/virt-install

- [7] Red Hat Cluster Suite:
https://access.redhat.com/documentation/es-ES/Red_Hat_Enterprise_Linux/6/html/Cluster_Suite_Overview/ch.gfscs.cluster-overview-CSO.html

- [8] Red Hat Linux Virtual Server:
https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/5/html/Virtual_Server_Administration/index.html

- [9] Piranha:
http://www.linuxvirtualserver.org/docs/ha/piranha.html

- [10] Prestashop:
http://doc.prestashop.com/display/PS16/Installing+PrestaShop

13. Anexos.

13.1. Pliego de condiciones técnicas.

En este apartado se especifica los aspectos principales que debería tener un pliego técnico para contratar un desarrollo análogo al abordado en este trabajo.

Objetivo.

El presente contrato tiene la finalidad de contratar los servicios para crear la infraestructura necesaria donde poder desplegar una tienda virtual con alta disponibilidad y balanceo de carga.

Presupuesto.

El presupuesto para la presente contratación es de 6955€ IGIC incluido que se reparte de la siguiente manera:

- 750€ correspondientes a 50 horas de tareas de planificación del sistema.
- 5.000€ correspondientes a 200 horas de tareas de desarrollo.
- 750€ correspondientes a 50 horas de tareas de prueba.
- 455€ correspondientes al 7% de IGIC, de la suma de las dos cifras anteriores.

Especificaciones técnicas.

Para abordar este proyecto se requerirán los siguientes recursos:

- Hardware:
  - Un ordenador personal en uno de los laboratorios de investigación con acceso al sistema anfitrión.
  - Sistema anfitrión capaz de albergar siete máquinas virtuales y que tenga acceso a la cabina de discos e Internet.
  - Cabina de discos.

- Software:
  - CentOS Linux 6 será el sistema operativo que dará soporte a al software de alta disponibilidad y balanceo de carga, que en todo momento deberá estar bajo la licencia de software libre.
  - KVM será la plataforma de virtualización.
  - Conga se encargará de la creación y configuración del clúster de bases de datos en alta disponibilidad.
  - Piranha + ipvsadm serán usados para la creación y configuración del clúster de servidores web en alta disponibilidad.
  - Apache, PHP y Mysql terminarán de conformar la infraestructura necesaria para desplegar el servicio de tienda virtual Prestashop.

- De red: Será necesaria una infraestructura de red capaz de comunicar el ordenador del laboratorio con el sistema anfitrión y este último con la cabina de discos.

Requisitos de fiabilidad.

El desarrollo de la solución debe realizarse de tal forma que se garantice su fiabilidad y comportamiento estable. Concretamente se utiliza unos recursos con redundancia para que el sistema siga funcionando, incluso en el caso de que:

- Caída de uno de los servidores web de tienda virtual.
- Caída de uno de los servidores de bases de datos.
- Caída de uno de los balanceadores de carga.

Además esta redundancia permite garantizar una calidad de servicio de manera que las peticiones se repartan en todo momento entre los servidores web activos.

Plazo de Ejecución.

El contratista se compromete a realizar el trabajo en un plazo máximo de cinco meses que empezará a contar desde la firma del contrato y que cumplirá con lo establecido en este documento.

Penalización.

La no entrega en el tiempo establecido conllevará una penalización del 0.002% por día de retraso, no pudiendo superar el 15% del presupuesto final.

Documentación.

El contratista se compromete a generar toda la documentación necesaria que se entregará al contratante tanto en formato digital como una copia impresa encuadernada.

Confidencialidad de la información.

De conformidad a lo establecido por la Ley Orgánica 15/1999 de Protección de Datos de Carácter personal (LOPD) y del Real decreto 1720/2007 (RDLOPD) del Reglamento de desarrollo de la LOPD, las dos partes firmantes quedan expresamente obligados a mantener absoluta confidencialidad y reserva sobre cualquier dato, especialmente los de carácter personal.

Propiedad del resultado.

Tanto el sistema de información creado como la documentación generada pasarán a formar parte de la Escuela de Ingeniería Informática, quienes podrán ejercer su derecho de explotación sobre los mismos. La entrega se hará conforme al Proceso de Gestión de la Entrega definido por la Dirección de la Escuela de Ingeniería Informática.

13.2. Archivo de configuración del clúster de servidores web con balanceo de carga.

```
serial_no = 65
primary = 192.168.122.11
service = lvs
backup_active = 1
backup = 192.168.122.12
heartbeat = 1
heartbeat_port = 539
keepalive = 6
deadtime = 18
network = direct
debug_level = NONE
monitor_links = 0
syncdaemon = 1
syncd_iface = eth0
syncd_id = 0
virtual Balanceador {
     active = 1
     address = 192.168.122.50 eth0:1
     port = 80
     send = "GET / HTTP/1.0\r\n\r\n"
     expect = "HTTP"
     use_regex = 0
     load_monitor = none
     scheduler = rr
     protocol = tcp
     timeout = 6
     reentry = 15
     quiesce_server = 0
     server Server1 {
         address = 192.168.122.21
         active = 1
         port = 80
         weight = 1
     }
     server Server2 {
         address = 192.168.122.22
         active = 1
         port = 80
         weight = 1
     }
}
```

13.3. Archivo de configuración del clúster de servidor de base de datos en alta disponibilidad.

```
<?xml version="1.0"?>
<cluster config_version="15" name="ClusterBD">
   <clusternodes>
      <clusternode name="ServerBD1" nodeid="1"/>
      <clusternode name="ServerBD2" nodeid="2"/>
   </clusternodes>
   <cman expected_votes="1" two_node="1"/>
   <rm>
      <resources>
         <ip address="192.168.122.100/24" sleeptime="10"/>
         <clusterfs device="/dev/Grupo_Almacen/Volumen_Almacen" force_unmount="1"
fsid="4605" fstype="gfs2" mountpoint="/Almacen" name="GFS2"/>
         <script file="/etc/init.d/iscsi" name="Iscsi"/>
         <script file="/etc/init.d/clvmd" name="Clvmd"/>
         <script file="/etc/init.d/mysqld" name="Mysqld"/>
      </resources>
      <failoverdomains>
         <failoverdomain name="ClusterBD" nofailback="1" ordered="1" restricted="1">
            <failoverdomainnode name="ServerBD1" priority="1"/>
            <failoverdomainnode name="ServerBD2" priority="2"/>
         </failoverdomain>
         <failoverdomain name="ServerBD1" restricted="1">
            <failoverdomainnode name="ServerBD1"/>
         </failoverdomain>
         <failoverdomain name="ServerBD2" restricted="1">
            <failoverdomainnode name="ServerBD2"/>
         </failoverdomain>
      </failoverdomains>
      <service domain="ClusterBD" name="SClusterBD" recovery="relocate">
         <ip ref="192.168.122.100/24"/>
         <script ref="Mysqld"/>
      </service>
      <service domain="ServerBD1" name="SServerBD1" recovery="restart">
         <script ref="Iscsi">
            <script ref="Clvmd">
               <clusterfs ref="GFS2"/>
            </script>
         </script>
      </service>
      <service domain="ServerBD2" name="SServerBD2" recovery="restart">
         <script ref="Iscsi">
            <script ref="Clvmd">
               <clusterfs ref="GFS2"/>
            </script>
         </script>
      </service>
   </rm>
</cluster>
```

13.4. Script de generación de MACs.

```
#!/usr/bin/env python
#  -*- mode: python; -*-
print ""
print "New UUID:"
import virtinst.util ; print virtinst.util.uuidToString(virtinst.util.randomUUID())
print "New MAC:"
import virtinst.util ; print virtinst.util.randomMAC()
print ""
```

13.5. Archivo de configuración XML de la máquina virtual base.

```
<domain type='kvm'>
  <name>TFG_WebServer1</name>
  <uuid>f0fa1090-a325-e37f-664c-ad634d531683</uuid>
  <memory unit='KiB'>1048576</memory>
  <currentMemory unit='KiB'>1048576</currentMemory>
  <vcpu placement='static'>1</vcpu>
  <os>
    <type arch='x86_64' machine='rhel6.3.0'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
    <pae/>
  </features>
  <clock offset='utc'/>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>restart</on_crash>
  <devices>
    <emulator>/usr/libexec/qemu-kvm</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='raw' cache='none'/>
      <source file='/mvirt/egarcia/TFG_Balanceo/WebServer1.img'/>
      <target dev='vda' bus='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
    </disk>
    <disk type='block' device='cdrom'>
      <driver name='qemu' type='raw'/>
      <target dev='hdc' bus='ide'/>
      <readonly/>
      <address type='drive' controller='0' bus='1' target='0' unit='0'/>
    </disk>
    <controller type='usb' index='0'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x2'/>
    </controller>
    <controller type='ide' index='0'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x1'/>
    </controller>
    <interface type='network'>
      <mac address='00:16:3e:c8:09:e1'/>
      <source network='default'/>
      <model type='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
    </interface>
    <serial type='pty'>
      <target port='0'/>
    </serial>
    <console type='pty'>
      <target type='serial' port='0'/>
    </console>
    <input type='tablet' bus='usb'/>
    <input type='mouse' bus='ps2'/>
    <graphics type='vnc' port='-1' autoport='yes'/>
    <video>
      <model type='cirrus' vram='9216' heads='1'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>
    </video>
    <memballoon model='virtio'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
    </memballoon>
  </devices>
</domain>
```