Servidores Webs de Altas Prestaciones (2015)
Grado en Ingeniería Informática
Universidad de Granada

Laboratorio de Una Granja Web

Carlos de la Torre

16 de mayo de 2015

Índice

1. Resumen/Abstract

En este documento se pretende describir como se puede implementar un CPD con diferentes software Open Source y que se pueda utilizar sin ningún problema en un entorno de producción, osea, que sea capaz de dar los servicios necesarios de un CPD común, FTP, MAIL, WWW, DNS, sin que por ello se baje el rendimiento de mismo.
También es cierto que el laboratorio que se ha propuesto para la practica tiene pocos recursos y la capacidad de proceso que tiene es limitada, bastaría con añadir recursos a las diferentes maquinas para que la capacidad de procesamiento creciera sin ningún problema.
También se describirá, cual es la topología utilizada para el CPD, las diferentes configuraciones de los servidores, una breve descripción de que es lo que hace cada uno de los servidores, la metodología utilizada para realizar la practica, y el porque se ha diseñado el CPD de esa manera.

This paper aims to describe how you can implement a CPD with different Open Source software and can be used without problem in a production environment, that is, to be able to give the necessary services of a common CPD, FTP, MAIL, WWW, DNS, without this same performance is lowered.
It is also true that the laboratory has been proposed for the practice has few resources and processing power you have is limited, simply add resources to different machines for processing capacity to grow without any problem.
Also will be described, which is the topology used for the CPD, different server configurations, a brief description of what makes each of the servers, the methodology used for practice, and why we have designed the CPD that way.

2. Descripción del entorno de trabajo

Para poder explicar de la mejor manera posible como poder crear un CPD de bajo coste y que se pueda utilizar en un entorno de producción sin que afecte al rendimiento se ha optado por crear un laboratorio que se asemeje a un CPD real.
Para ello se han utilizado 6 maquinas virtuales que se encargaran de los diferentes servicios que tiene que alojar un CPD.

1. FIREWALL
2. DNS
3. DHCP
4. LOAD BALANCED
5. WWW
6. FTP
7. EMAIL (opcional)

Por supuesto que al ser un entorno virtual, y siendo solamente un laboratorio, hay algunas restricciones inherentes a dicho entorno, como por ejemplo, que desde el exterior de la red virtual solo se permite el acceso a solo un servidor web, que en este caso sera el balanceador de carga.
Otro aspecto a tener en cuenta en la configuración del laboratorio es que aunque se ha diseñado el mismo para tener una configuración de doble DMZ la parte de la red empresarial no se ha implementado en dicho laboratorio, ya que esta su implementación es trivial, y solo se deseaba simular la configuración de los servidores y el acceso de los mismo a Internet.

2.1. Descripción de la maquina física

Para mondar dicho laboratorio se ha utilizado una maquina que denominaremos host con la siguiente configuración, tanto de hardware como de software:

2.1.1. Hardware

1. Portátil Phoenix
2. Placa Base: Pegatron Corporation model: H36Y
3. Procesador: Intel® Core™ i5 CPU M430 @ 2.27GHz 4 nucleos x64_86
4. Memoria: Transcend 2 x JM1066KSN-4Gb 1066MHz (0.9ns de acceso)
5. Gráfica: Intel® Corporation VGA Compatible
6. Disco Duro (OS): Samsung SSD 840 250Gb
7. Disco Duro (VM): Seagate ST1000LM024 HN-M 1TB
8. Red: Qualcomm Atheros AR8131 Gigabit Ethernet
9. Wifi: Qualcomm Atheros AR9285 Wireless Network Adapter

2.1.2. Software

1. Sistema Operativo: Fedora Spin 21 x64_86
2. Kernel: 3.18.9-100.fc20.x86_64
3. Gestor Gráfico: Qt 4.8.6, KDE: 4.14.6
4. HyperVisor: VMware Workstation 11.0.0 build-2305329

En la configuración de esta maquina se ha implementado un firewall que lo único que hace es securizar el acceso a la maquina física y enrrutar los diferentes puertos a las diferentes plataformas de gestión en las maquinas virtuales así como el propio acceso a los diferentes servicios de la granja WEB.

2.2. Descripción de las maquinas virtuales

Como las maquinas virtuales utilizadas parten de una misma base, describiremos la base desde la cual se han clonado todas y a continuación iremos agregando o quitando propiedades a las diferentes maquinas para que se amolden a las necesidades de los servicios que tienen que prestar dentro del CPD.

2.2.1. Maquina Virtual Base

Datos de la maquina base:

Hardware		**Software**	
Procesador:	1 x Intel® Core™ i5 @ 2.27GHz	**Sistema Operativo:**	Centos 7 x64_86
Memoria:	384 Mb	**Kernel:**	
Disco Duro:	8Gb SATA	**Gestor Gráfico:**	Sin gestor
Red 1:	VmNet8 NAT	**Addons:**	
Red 2:	VmNet1 Host Only	Webmin, Apache, MySQL, Issue panel	

2.2.2. Maquina DNS

En esta maquina como se utilizan los servicios de DNS, DHCP y Firewall norte ninguno de ellos son servicios que necesiten demasiados recursos por lo que se ha hecho es bajar la configuración de la memoria hasta los 256 Mb y solo una tarjeta de red, y por supuesto se ha añadido el software necesario para poder configurar ambos servicios, aparte se han añadido también los módulos necesarios de WebMin.

Hardware		**Software**	
Procesador:	1 x Intel® Core™ i5 @ 2.27GHz	**Sistema Operativo:**	Centos 7 x64_86
Memoria:	256 Mb	**Kernel:**	3.10.0-229.1.2.el7.x86_64
Disco Duro:	8Gb SATA	**Gestor Gráfico:**	Sin gestor
Red 1:	VmNet1 Host Only	**Addons:**	
Red 2:	VmNet8 NAT	Webmin, Issue panel, DNS, DHCP, IPTABLES	

2.2.3. Maquina BALANCEADOR

Esta maquina sera la encargada de repartir entre los servidores web la carga de trabajo que llegue desde internet, no es mas que un balanceador de carga por software.

Hardware		**Software**	
Procesador:	1 x Intel® Core™ i5 @ 2.27GHz	**Sistema Operativo:**	CentOS 7.1 x64_86
Memoria:	256 Mb	**Kernel:**	3.10.0-229.1.2.el7.x86_64
Disco Duro:	8Gb SATA	**Gestor Gráfico:**	Sin gestor
Red 1:	VmNet1 Host Only	**Addons:**	Webmin, Issue panel, NGINX Proxy

2.2.4. Maquinas WEB1 y WEB2

Estas maquinas serán las encargadas de servir las paginas web que se desarrollan en la asignatura de Tecnologías Webs, para las practicas de esta asignatura, hay previsto hacer una tienda web, tanto de manera estática como de manera dinámica utilizando una base de datos de productos y de usuarios que accederán a la gestión de dicha tienda.
Para poder mostrar un entorno lo mas real posible se integraran estas paginas junto con las bases de datos en el CPD portátil del laboratorio de estudio.

Hardware		**Software**	
Procesador:	1 x Intel® Core™ i5 @ 2.27GHz	**Sistema Operativo:**	CentOS 7.1 x64_86
Memoria:	384 Mb	**Kernel:**	3.10.0-229.1.2.el7.x86_64
Disco Duro:	8Gb SATA	**Gestor Gráfico:**	Sin gestor
Red 1:	VmNet1 Host Only	**Addons:**	Webmin, Apache, Issue panel, FTP

2.2.5. Maquinas DB1 y DB2

Estas maquinas son las encargadas de mantener en funcionamiento las base de datos que alimenta la aplicación web, al ser un entorno de prueba tiene una memoria RAM escasa, pero como son maquinas

virtuales, bastaría con mudar la maquina a un servidor mas grande y ampliar dicha memoria.

Hardware		**Software**	
Procesador:	1 x Intel® Core™ i5 @ 2.27GHz	**Sistema Operativo:**	CentOS 7.1 x64_86
Memoria:	384 Mb	**Kernel:**	3.10.0-229.1.2.el7.x86_64
Disco Duro:	8Gb SATA	**Gestor Gráfico:**	Sin gestor
Red 1:	VmNet1 Host Only	**Addons:**	
Webmin, Issue panel, MariaDB, PostgreSQL, phpMyAdmin, phpPgAdmin			

2.3. Entorno DMZ

Como ya se ha explicado varias veces en clase esta parte de la topología de red, denominada así por su similitud con una zona de un conflicto armado, se utiliza para asegurar aquella zona de la topología, la cual aun teniendo sus propios sistemas de seguridad, se aísla de la red de manera física aportando así un extra en dicha seguridad.
Para este laboratorio, lo que haremos es montar solamente la parte que se ve desde internet, y por lo tanto tiene un escenario particular.
Empezando por que el FW norte es de tipo sofware y tiene recursos compartidos con el servidor de DNS y DHCP, por supuesto esto en un entorno real no lo montaremos así nunca puesto que supone un grabe riesgo para la seguridad de nuestra red.
Aparte de todo esto tan obvio, en la red empresarial no he puesto ningún servidor de dominio por lo tanto la gestión de los servidores se tendría que realizar a través de las direcciones IP de cada uno de ellos, aunque para resolver esto sería tan sencillo como crear un servidor de dominio local para que la red empresarial pudiera acceder a ellos por su nombre, esta situación, en la que tenemos que implementar dos servidores de dominio, se produce a causa de la configuración de la doble DMZ, ya que un mismo servidor de dominio no puede resolver dos direcciones IP diferentes para un mismo nombre de dominio. Por lo tanto tendremos que tener dos dominios diferentes, y aunque se pueden configurar varios dominios en un mismo servidor, en nuestro laboratorio no se podría realizar esa configuración por la manera en la que se han conectado los equipos a la red.

2.4. Topología completa de la Red Física

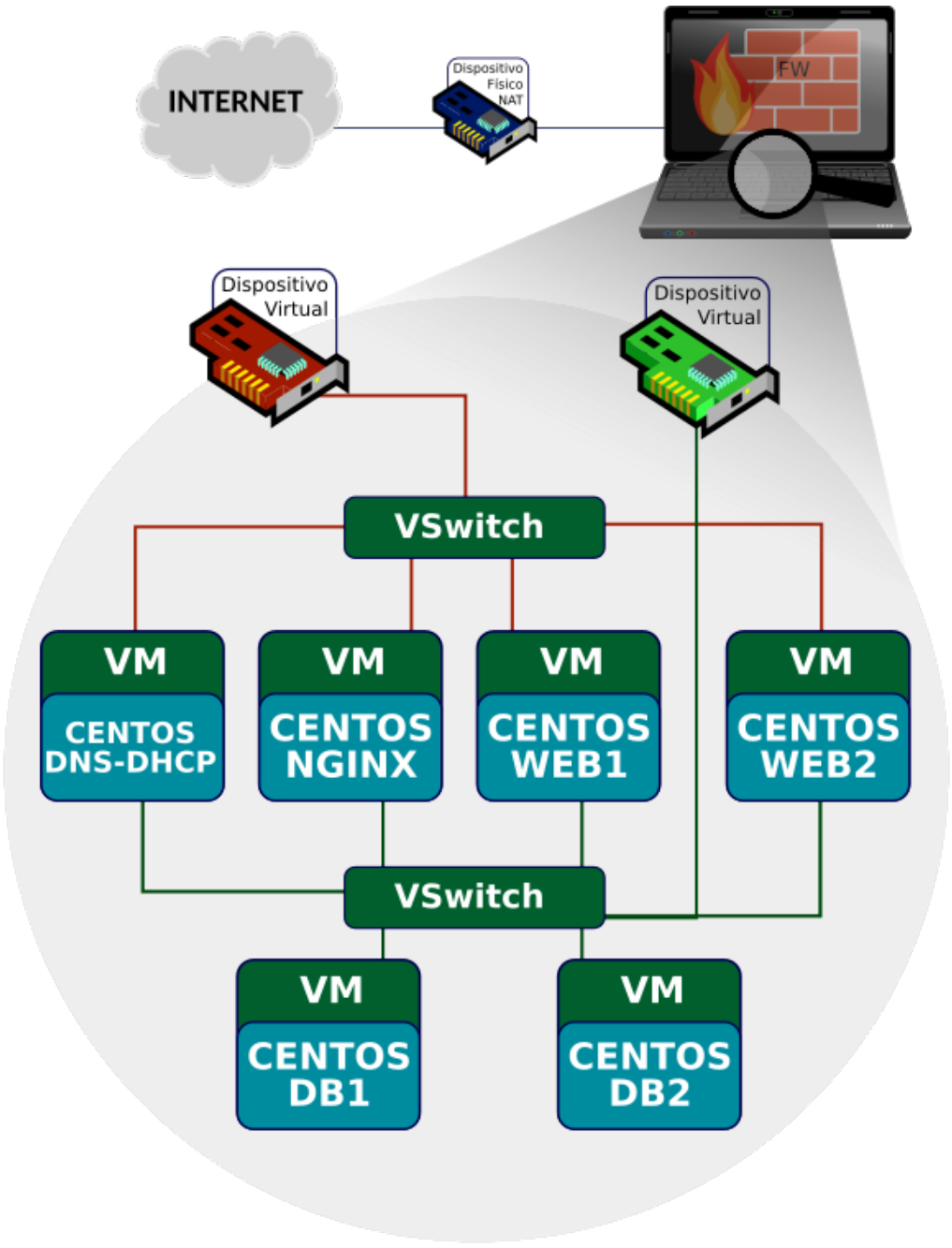


Figura 2.1: Topología Física

2.5. Topología completa de la Red Lógica

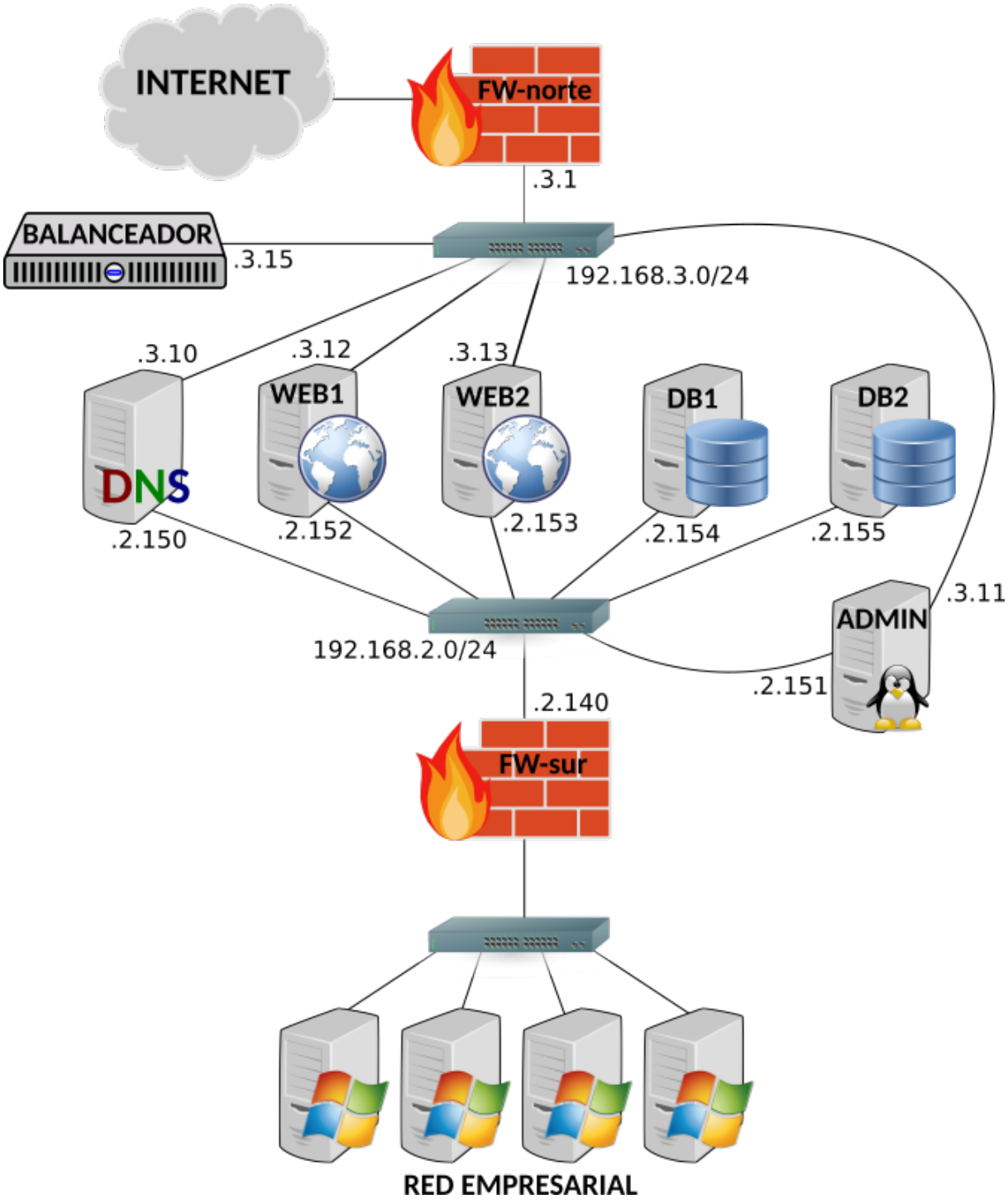


Figura 2.2: Topología Lógica

3. Descripción de la parte del Firewall

Como ya he dicho con anterioridad en este laboratorio vamos a utilizar una configuración con dos firewalls (norte y sur), el firewall norte será el propio portátil que será el que tenga la entrada desde internet y luego tendremos otra maquina virtual que será el firewall sur, ambas maquinas implementan iptables para hacer las veces de firewall.
Las reglas básicas de configuración en ambos firewalls son que bloqueamos todo el tráfico entrante y saliente, y a partir de ahí configuramos los protocolos y los puertos necesarios para poder dar a los usuarios los diferentes servicios que tenemos integrados en nuestro CPD virtual.

3.1. Alternativas a IpTables

Aún que en este laboratorio vamos a utilizar una configuración de IpTables en su forma mas rudimentaria, osea, a través de la linea de comandos, también tenemos varias formas de poder configurar de manera gráfica las reglas y las tablas de iptables.
Una de las alternativas mas conocidas para configurar iptables es a través de interfaz web, en este laboratorio también haremos uso de **WebMin**[9] no solo para configurar iptables sino muchos de los parámetros de nuestros servidores virtuales.
Otra de las posibles alternativas, es utilizar una distribución de linux completa dedicada única y exclusivamente a ser usada como firewall de red: **IPCOP**[6], por supuesto también es una solución software pero, al estar dedicada exclusivamente a un cometido especifico sus resultados son mas que aceptables.

3.2. Posibilidades de Hardware para el Firewall

Una de las posibilidades que existen en el mercado para implementar un firewall por hardware es utilizar un componente especifico de una de las marcas mas conocidas en el mundo del networking como es CISCO, en este caso utilizaríamos uno de sus productos también muy conocido como el ASA[14], este componente es un firewall por hardware pero que por supuesto no solamente hace las funciones de firewall si no que tiene muchísimas mas opciones, como, VPN múltiples, Análisis automático del trafico de red, Previene el acceso no autorizado, Clusterizado de infraestructura de red, etc...
En una breve búsqueda hecha por algunas paginas de compra online, los precios que he conseguido encontrar para este tipo de aparatos oscilan entre los 290 € del modelo ASA5005-K9 hasta los 13500 € del modelo ASA5555-K9.

Figura 3.1: Diferentes modelos de ASA

3.3. Ventajas y desventajas, opiniones, conclusiones

Por supuesto cuando utilizamos este tipo de configuración en la que todo nuestros dispositivos son de tipo software tenemos una clara desventaja frente a un entorno en donde tenemos equipos hardware dedicados única y exclusivamente a un cometido particular, de todos es sabido que si tenemos un aparato que hace una determinada tardea será mas rápido que un programa que hace esa misma tarea.
Claro que esto tiene la ventaja que podemos reutilizar cualquier viejo ordenador para usarlo como un dispositivo para un fin especifico, y por supuesto un ahorro considerable, ya que los dispositivos hardware casi siempre serán mas caros que una solución software.
Personalmente he estado utilizado durante muchos años (8) la solución software **IPCOP**, en un entorno de producción con unos resultados muy buenos ya que es bastante configurable y modular con apoyo de la comunidad. Esta claro que siempre que el presupuesto lo permita tendremos que usar soluciones hardware pero cuando queremos ajustarnos a precios competitivos y según en que entornos podemos decantarnos por este tipo de opciones.

4. Descripción de la parte del Balanceador de Carga

Como su propio nombre indica este componente de la infraestructura de red sirve para poder tener un punto de entrada al entorno de red pero que a su vez se encarga de repartir el trabajo entre los diferentes servidores que dan un determinado servicio, por ejemplo en nuestro laboratorio, utilizaremos un balanceador de carga para el servicio web, osea, que tendremos varios servidores web que se encargaran de servir paginas web y según requiera la carga de trabajo se utilizaran varios o solamente uno de ellos para servir las mismas.
En este laboratorio utilizaremos un balanceador por software, mas concretamente **NGINX**,[13] que aunque es mas conocido por ser un servidor de páginas web también se puede configurar como balanceador de carga.
En la imagen de la topología de red (figura **??**) se muestra en que lugar suelen estar posicionados estos dispositivos, ya que suelen recibir las peticiones de los clientes desde internet, casi siempre estarán colocados a continuación del firewall norte.

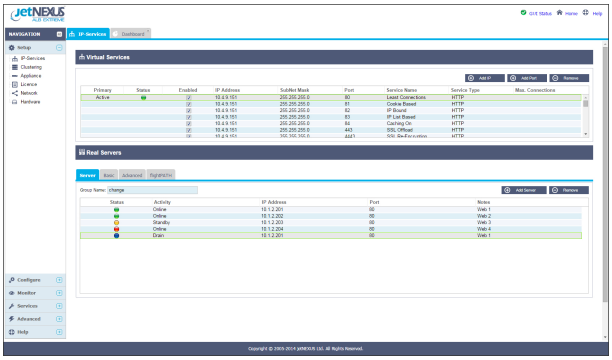
4.1. Alternativas a NGINX

Unas de las posibles alternativas podría ser **HAProxy**,[11] como en su propia página lo presentan, este podría ser una solución bastante viable para entornos de producción que necesiten una alta carga de trabajo, también es una solución por software que da muy buenos resultados, la configuración del mismo no suele ser muy compleja, y los requisitos de hardware no son muy elevados, 1 Gb de memoria y un procesador normalito.

4.2. Posibilidades de Hardware para el balanceador

Por supuesto hay muchas posibilidades para utilizar un producto hardware como balanceador de carga, pero realizando una pequeña búsqueda por internet uno de los que mas suenan es: **NetScaler de Citrix**[2] este componente hardware es bastante caro, aunque, si nos paramos a pensar un momento la carga de trabajo que tiene que soportar dicho componente es natural que la electromecánica de red que se desarrolle tiene que ser robusta y por consiguiente cara.
Otra de las posibilidades que he encontrado en internet es: **jetNexus**[7] aunque este último no he conseguido encontrar su precio, las características técnicas de dicho dispositivo son impresionantes, y cuenta con la certificación de bastantes fabricantes de networking que muestran su apoyo para la implementación del mismo.

(a) Hardware de jetNexus



(b) Interfaz gráfica de configuración de jetNexus

Figura 4.1: Hardware y software de jetNexus

4.3. Ventajas y desventajas, opiniones, conclusiones

Ls ventajas para estas configuraciones están muy claras, con esta forma de configura nuestro CPD nos da la posibilidad de incrementar el nivel de computo que tenemos en la granja web fácilmente pues bastaría con integrar mas servidores en la granja web y añadirlo a la configuración de NGINX para que este ultimo servidor sirviera las paginas que tenemos alojado en nuestro CPD.
Una desventaja que se ve claramente es que en este punto tenemos un *'cuello de botella'*, osea, que nuestra granja web depende directamente de la capacidad de transferencia y computo de nuestro balanceador de carga, osea, que hay que tener mucho cuidado a la hora de dimensionar nuestra infraestructura de red

para no encontrarnos con este problema.
Personalmente pienso que la solución implementada en este laboratorio se podría implementar sin problemas mientras nuestra carga de trabajo fuese baja-media, pero en el momento que nuestro CPD tuviera una carga alta de trabajo esta solución seria insuficiente.

5. Descripción de la parte del Servidor DNS[1]

El servidor DNS utiliza una base de datos distribuida y jerárquica que almacena información asociada a nombres de dominio en redes como Internet. Aunque como base de datos el DNS es capaz de asociar diferentes tipos de información a cada nombre, los usos más comunes son la asignación de nombres de dominio a direcciones IP y la localización de los servidores de correo electrónico de cada dominio.

5.1. Tipos de Servidores

1. Primarios o maestros: Guardan los datos de un espacio de nombres en sus base de datos.

2. Secundarios o esclavos: Obtienen los datos de los servidores primarios a través de una transferencia de zona.

3. Locales o caché: Funcionan con el mismo software, pero no contienen la base de datos para la resolución de nombres. Cuando se les realiza una consulta, estos a su vez consultan a los servidores DNS correspondientes, almacenando la respuesta en su base de datos para agilizar la repetición de estas peticiones en el futuro continuo o libre.

5.2. Alternativas a BIND[12]

Dnsmasq es una de las posibles soluciones a bind, aunque por supuesto no es comparable, dnsmasq esta programado en C es muy veloz y también es muy pequeño, ha sido un elemento decisivo para el desarrollo de las nuevas tecnologías en smartphones, su configuración no es muy compleja y es un software fiable po su gran aceptación en todo tipo de plataformas.

5.3. Ventajas y desventajas, opiniones, conclusiones

Bueno las ventajas de este tipo de servicio es obvia, si tienes mala memoria este es tu servicio, puesto que solo te tienes que acordar de un solo dominio para poder gestionar toda la red, ahora bien por experiencia es un servicio, con un grado de dificultad medio alto a la hora, tanto de configurar como de gestionar, puesto que las pruebas de concepto realizadas con un sistema operativo Windows, y aunque este es mas sencillo de configurar, basta un mínimo cambio en la configuración del servidor DNS para que de problemas de replicación o algún otro parecido.

Mi opinión personal es que es un servicio necesario en una red media grande, pero que se tiene que desplegar correctamente desde el principio si no queremos tener dolores de cabeza después, ademas de eso siempre es MUY aconsejable tener el dominio replicado en un servidor esclavo.

Conclusión si no tienes mas remedio despliegalo, pero si puedes, por que tu configuración no te lo exige, despliegalo.

6. Descripción de la parte del Servidor Apache[5]

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que Behelendorf quería que tuviese la connotación de algo que es firme y enérgico pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la

preocupación de su grupo era que llegasen las empresas y çivilizasen.el paisaje que habían creado los primeros ingenieros de internet. Además Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. En inglés, a patchy server (un servidor "parcheado") suena igual que Apache Server.

6.1. Alternativa NGINX[13]

NGINX Originalmente, nginx fue desarrollado para satisfacer las necesidades de varios sitios web de Rambler[10] que recibían unas 500 millones de peticiones al día en septiembre de 2008.
De acuerdo con el estudio de Netcraft, Netcraft's Jul 2014 Web Server Survey, nginx es el segundo servidor web más usado en dominios activos (14,35 %) superando a Microsoft Information Server.
Además, pasó la marca de ser usado en más de 100 millones de sitios.

6.2. Ventajas y desventajas, opiniones, conclusiones

Es openSource, tiene una comunidad enorme a sus espaldas, esta para todas las plataformas, es robusto, modular, y un sin fin de cosas mas, ahora bien este tipo de servidor esta recomendado para según que condiciones de trabajo, hay que conocer bien sus configuraciones para poder sacarle el máximo partido y tiene muchísimas configuraciones posibles. En mi opinión es un servidor web mas que probado mas que confiable y de renombre que se puede usar sin demasiados problemas en la mayoría de sitios web, pero hay que tener en cuenta que hay mas opciones aparte de este servidor.

7. Descripción de la parte del Servidores Base de Datos

MariaDB es un sistema de gestión de bases de datos derivado de MySQL con licencia GPL. Es desarrollado por Michael (Monty) Widenius (fundador de MySQL) y la comunidad de desarrolladores de software libre. Introduce dos motores de almacenamiento nuevos, uno llamado Aria -que reemplaza con ventajas a MyISAM- y otro llamado XtraDB -en sustitución de InnoDB. Tiene una alta compatibilidad con MySQL ya que posee las mismas órdenes, interfaces, APIs y bibliotecas, siendo su objetivo poder cambiar un servidor por otro directamente Este SGBD surge a raíz de la compra de Sun Microsystems -compañía que había comprado previamente MySQL AB - por parte de Oracle. MariaDB es un fork directo de MySQL que asegura que permanecerá una versión de este producto con licencia GPL. Monty decidió crear esta variante porque estaba convencido de que el único interés de Oracle en MySQL era reducir la competencia que MySQL daba al mayor vendedor de bases de datos relacionales del mundo que es Oracle.

7.1. Alternativas a MariaDB[8, 16]

PostgreSQL[3, 4, 15] es un Sistema de gestión de bases de datos relacional orientado a objetos y libre, publicado bajo la licencia BSD.
Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre y/o apoyados por organizaciones comerciales. Dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group).

7.2. Ventajas y desventajas, opiniones, conclusiones

En este apartado las ventajas y desventajas están un tanto difusas puesto que el uso de un SGDB, es relativo y muy subjetivo al uso que se le vaya a dar en la aplicación que estamos desarrollando, incluso con un mismo SGDB, podemos obtener diferentes respuestas según en la manera que este este configurado.
Esta claro que siendo MariaDB un fork directo de la ya tan conocida MySQL la opinión será buena puesto que da capacidad a los desarrolladores a poder seguir manteniendo sus viejas aplicaciones web con soporte absoluto en MySQL gracias a este fork.

8. Metodología de trabajo

Para desarrollar este laboratorio he seguido una serie de normas básicas, para no perderme en la configuración de tantas IPs, y maquinas diferentes, lo primero que hice fue configurar correctamente una maquina virtual que me sirviera como plantilla base, así pues genere una maquina virtual limpia e instale

un sistema operativo Centos 7.1, después de eso instale el software de gestión WebMin para que fuese mas sencillo poder configurar las diferentes opciones del servidor.

Esta claro que todos los servidores de un entorno de granja web van a tener unos servicios mínimos y comunes entre ellos, como pueden ser SSH, NTP, NETWORK, Gráficas de estadísticas, etc... pues después de la instalación se configuraron todos estos parámetros para luego poder clonar la maquina que el resto de servidores tuvieran la misma base, por supuesto con diferentes IPs.

Para que la gestión de los servidores sea mas sencilla a través de consola he generado una clave privada y la he copiado ha esta plantilla así de esa manera cuando queramos conectarnos a cualquiera de los servidores podremos hacerlo de forma segura a través del protocolo SSH.

En un entorno de red también es importante tener sincronizados los relojes de los servidores, por este motivo también he configurado el protocolo NTP para que así todos los servidores tengan la misma hora, de esa manera si hay que realizar cualquier auditoria, los ficheros de bitácora (logs), guardaran los datos con el mismo orden y por lo tanto será mas fácil detectar cualquier problema.

8.1. Configuración General

Algunos de los ficheros de configuración de todos los servidores se muestran a continuación, por supuesto con pequeñas diferencias entre ellos puesto que al ser maquinas diferentes no podrían tener los mismos parámetros.

Configuración de *ssh* /etc/ssh/sshd_config

```
# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#

# The default requires explicit activation of protocol 1
#Protocol 2

HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key

# Logging
# obsoletes QuietMode and FascistLogging
#SyslogFacility AUTH
SyslogFacility AUTHPRIV
#LogLevel INFO

# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
AuthorizedKeysFile  .ssh/authorized_keys

# To disable tunneled clear text passwords, change to no here!
#PermitEmptyPasswords no
PasswordAuthentication yes

# Change to no to disable s/key passwords
#ChallengeResponseAuthentication yes
ChallengeResponseAuthentication no

# GSSAPI options
#GSSAPIAuthentication no
GSSAPIAuthentication yes
```

```
#GSSAPICleanupCredentials yes
GSSAPICleanupCredentials yes
#GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no

# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
# WARNING: 'UsePAM no' is not supported in Red Hat Enterprise Linux and may cause several
# problems.
#UsePAM no
UsePAM yes

X11Forwarding no
UsePrivilegeSeparation sandbox    # Default for new installations.

# no default banner path
#Banner none

# Accept locale-related environment variables
AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL LANGUAGE
AcceptEnv XMODIFIERS

# override default of no subsystems
Subsystem sftp  /usr/libexec/openssh/sftp-server

# Uncomment this if you want to use .local domain
#Host *.local
# CheckHostIP no
```

sshd_config

Configuración de *issues* /etc/issue.sh & /etc/issue.default

```
#!/bin/bash                                          |  Servidor #host#   Practicas de SWAP
                                                     | ------------------------------------
ISSUE_FILE_DEFAULT=/etc/issue.default                | Configurado por:   Carlos de la Torre
ISSUE_FILE=/etc/issue                                | Sistema:           #os#
OS_FILE=/etc/centos-release                          | Kernel:            \r on an \m (\l)
                                                     | IPs acceso:        #ips#
cp $ISSUE_FILE_DEFAULT $ISSUE_FILE                   |
sed -i -e s/#os#/"$(cat $OS_FILE)"/g $ISSUE_FILE     |
sed -i -e s/#host#/$(hostname)/g $ISSUE_FILE         |
sed -i -e s/#ips#/"$(hostname -I)"/g $ISSUE_FILE     |
```

issue.default

Configuración de *network* /etc/sysconfig/network-scripts/ifcfg-eno16777736 & ifcfg-eno33554944

```
        IP Fija Verde              IP Fija Roja             IP Dinámica

                              | GATEWAY=192.168.3.1     | GATEWAY=192.168.3.1
NAME=""                       | NAME=""                 | NAME=""
BOOTPROTO=none                | BOOTPROTO=none          | BOOTPROTO=dhcp
NM_CONTROLLED=no              | NM_CONTROLLED=no        | NM_CONTROLLED=no
MACADDR=""                    | MACADDR=""              | MACADDR=""
IPV6INIT=no                   | IPV6INIT=no             | IPV6INIT=no
DEVICE=eno16777736            | DEVICE=eno33554944      | DEVICE=eno16777736
MTU=""                        | MTU=""                  | MTU=""
NETMASK=255.255.255.0         | NETMASK=255.255.255.0   | NETMASK=255.255.255.0
BROADCAST=192.168.50.255      | BROADCAST=0.0.0.255     | BROADCAST=192.168.50.255
IPADDR=192.168.2.150          | IPADDR=192.168.3.10     | IPADDR=""
NETWORK=192.168.2.0           | NETWORK=192.168.3.0     | NETWORK=""
ONBOOT=yes                    | ONBOOT=yes              | ONBOOT=yes
```

ifcfg–enox

8.2. Configuración especifica del DNS y DHCP

En el laboratorio vamos a intentar reproducir lo mas fielmente posible un entorno empresarial por lo tanto este debe de tener un servidor DHCP, así que para poder aprovechar los recursos de los servidores al máximo nivel, vamos a utilizar uno de los servidores para dar servicio de dos clases, que son: DHCP y DNS.

Así pues en este servidor tenemos que tener dos redes claramente definidas, una será nuestra red verde que será a la que se conectara nuestra red empresarial, y la otra será nuestra red roja que será la que se exponga a internet y de servicio a nuestros clientes externos.

Teniendo en cuenta estas consideraciones, tendremos que crear dos subredes tal y como se muestra en la topología figura 2.1, por supuesto nuestro servidor de DHCP-DNS será el único que tenga las direcciones IP colocadas de forma manual puesto que este es el que tiene que distribuir las IPs en nuestra red.

Después de esto reservaremos IPs para los distintos servidores que intervienen en nuestra red de esta manera nos aseguramos que utilizando la misma tarjeta de red tendremos el mismo servicio que le asignemos a esa IP. Así pues si uno de nuestros servidores WEB le asignamos mediante DHCP la IP 192.168.2.152/24 sabemos que esa IP siempre será un servidor WEB, a no ser que cambiemos de tarjeta de red, en cuyo caso lo único que tenemos que hacer es antes de arrancar el servidor sustituir la dirección MAC de la tarjeta de red nueva en la reserva de nuestro servidor DHCP.

Teniendo en cuenta que vamos a tener 2 subredes, es lógico tener 2 dominios, unos que será el que gestione nuestra red verde: **midominio.test** y otro que será **midominio.pub** que estará visible para nuestros usuarios desde internet (red roja). En este laboratorio como no es una situación realmente necesaria no incluiremos en la configuración del DNS las zonas inversas de cada uno de los dominios, pero si utilizaremos los registros mas comunes del servidor de DNS como pueden ser los A, y los CNAME[1].

Configuración de *dhcpd* /etc/dhcp/dhcpd.conf

```
max-lease-time 300;
default-lease-time 60;
#
# DHCP Server Configuration file.
#   see /usr/share/doc/dhcp*/dhcpd.conf.example
#   see dhcpd.conf(5) man page
# Esta es al RED Empresarial
shared-network RedLocal {
  # RedGranja
  subnet 192.168.2.0 netmask 255.255.255.0 {
    option domain-name-servers 192.168.2.150, 8.8.8.8;
    option routers 192.168.2.1;
    authoritative;
    range 192.168.2.30 192.168.2.100;
    # VISIBLES LOCAL
    group {
      # Portátil desde local
      host portatil-verde {
        option routers 0.0.0.0;
        hardware ethernet 00:50:56:C0:00:01;
        fixed-address 192.168.2.1;
        }
      # Servidor DB1 desde local
      host DB1-verde {
        hardware ethernet 00:0c:29:0b:c8:17;
        fixed-address 192.168.2.154;
        }
      # Servidor DB2 desde local
      host DB2-verde {
        hardware ethernet 00:0c:29:ad:77:9b;
        fixed-address 192.168.2.155;
        }
      # Servidor Web 1 desde local
```

```
      host WEB1-verde {
        hardware ethernet 00:0C:29:24:59:21;
        fixed-address 192.168.2.152;
        }
      # Servidor Web2 desde local
      host WEB2-verde {
        hardware ethernet 00:0C:29:E1:5C:9E;
        fixed-address 192.168.2.153;
        }
      }
    }
  }
# Esta es la RED por donde se accede desde Internet
shared-network RedExterna {
  # RedInternet
  subnet 192.168.3.0 netmask 255.255.255.0 {
    option domain-name-servers 192.168.3.10, 8.8.8.8;
    option routers 192.168.3.1;
    range 192.168.3.30 192.168.3.200;
    # VISIBLES INTERNET
    group {
      # Balanceador de Carga en Internet
      host LB-roja {
        hardware ethernet 00:0C:29:A8:78:07;
        fixed-address 192.168.3.15;
        }
      # Servidor WEB1 desde Internet
      host WEB1-roja {
        fixed-address 192.168.3.12;
        hardware ethernet 00:0C:29:24:59:2B;
        }
      # Portátil desde Internet
      host portatil-roja {
        option routers 0.0.0.0;
        hardware ethernet 00:50:56:C0:00:02;
        fixed-address 192.168.3.1;
        }
      # Servidor WEB2 desde internet
      host WEB2-roja {
        hardware ethernet 00:0C:29:E1:5C:A8;
        fixed-address 192.168.3.13;
        }
      }
    }
  }
```

dhcpd.conf

Configuración de *named* /etc/named.conf

```
//
// named.conf
//
// Provided by Red Hat bind package to configure the ISC BIND named(8) DNS
// server as a caching only nameserver (as a localhost DNS resolver only).
//
// See /usr/share/doc/bind*/sample/ for example named configuration files.
//
options {
        listen-on port 53 {
                127.0.0.1;
                192.168.2.150;
                192.168.3.10;
                };
        listen-on-v6 port 53 { none; };
        directory       "/var/named";
        dump-file       "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        allow-query      { localhost; 0.0.0.0/0;};
        allow-transfer { localhost; 0.0.0.0/0;};
        /*
         - If you are building an AUTHORITATIVE DNS server, do NOT enable recursion.
         - If you are building a RECURSIVE (caching) DNS server, you need to enable
```

```
25              recursion.
26           - If your recursive DNS server has a public IP address, you MUST enable access
27             control to limit queries to your legitimate users. Failing to do so will
28             cause your server to become part of large scale DNS amplification
29             attacks. Implementing BCP38 within your network would greatly
30             reduce such attack surface
31         */
32         recursion yes;
33
34         dnssec-enable yes;
35         dnssec-validation yes;
36         dnssec-lookaside auto;
37
38         /* Path to ISC DLV key */
39         bindkeys-file "/etc/named.iscdlv.key";
40
41         managed-keys-directory "/var/named/dynamic";
42
43         pid-file "/run/named/named.pid";
44         session-keyfile "/run/named/session.key";
45         forwarders {
46                 8.8.8.8;
47                 8.8.4.4;
48                 };
49         forward first;
50 };
51 logging {
52         channel default_debug {
53                 file "data/named.run";
54                 severity dynamic;
55         };
56 };
57 zone "." IN {
58         type hint;
59         file "named.ca";
60 };
61 include "/etc/named.rfc1912.zones";
62 include "/etc/named.root.key";
63
64 zone "midominio.test" {
65         type master;
66         file "/var/named/midominio.test.hosts";
67         };
68
69 zone "midominio.pub" {
70         type master;
71         file "/var/named/midominio.pub.hosts";
72         };
73 server 8.8.8.8 {
74         };
75 server 8.8.4.4 {
76         };
```

named.conf

8.3. Configuración especifica del Balanceador

Este es uno de los servidores menos pesados, aunque si es cierto que según las circunstancias será uno de los que tenga que soportar mayor carga de trabajo, puesto que será el encargado de distribuir las peticiones provenientes de internet a los diferentes servidores WEB.

La configuración de este servidor es muy sencilla puesto que solamente tendremos que configurar NGINX como balanceador de carga, y para eso lo único que tenemos que hacer es utilizar el fichero que viene a continuación y sustituirlo en la configuración por defecto de nginx.

Comentar también que aunque este servidor se podría integrar con algún otro servidor de baja carga, no lo haremos de esta manera, puesto que este es la puerta de entrada a nuestra red, así pues si tenemos una intrusión o bien tenemos un desastre en este servidor basta con sustituirlo, y así evitaremos cualquiera de las dos contingencias.

Configuración de *nginx* /etc/nginx/nginx.conf

```
events {
    worker_connections  1024;
}
http {
     upstream apaches {
          ip_hash;
          server 192.168.3.12; #max_fails=3 fail_timeout=5s;
          server 192.168.3.13;
          keepalive 10;
     }
     server{
         listen 80;
         server_name m3lb;
         access_log /var/log/nginx/m3lb.access.log;
         error_log /var/log/nginx/m3lb.error.log;
         root /var/www/;
         location /
         {
             proxy_pass http://apaches;
             proxy_set_header Host $host;
             proxy_set_header X-Real-IP $remote_addr;
             proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
             proxy_http_version 1.1;
             proxy_set_header Connection "";
         }
     }
}
```

nginx.conf

8.4. Configuración especifica de Apache

En la parte del servidor web de este laboratorio no hemos realizado configuraciones muy exhaustivas, ya que hay numerosa literatura en Internet, pero una de las tareas que si he realizado es utilizar el servidor web1 como principal y el servidor web2 como esclavo, así de esta manera cuando modifiquemos algún fichero en la pagina web que estamos sirviendo, tendremos los dos servidores de páginas sirviendo los mismos documentos.
Este proceso lo hemos realizado con la herramienta rsync y con la herramienta cron, así pues esta ultima se encarga de comprobar periódicamente que los ficheros coinciden en su tamaño y formato y la primera se encarga de realizar la copia si hiciera falta.

Configuración de *httpd*

```
ServerRoot "/etc/httpd"

#Listen 12.34.56.78:80
Listen *:80

# Example:
# LoadModule foo_module modules/mod_foo.so
#
Include conf.modules.d/*.conf

User apache
Group apache

ServerAdmin root@localhost

#ServerName www.example.com:80

#
# Deny access to the entirety of your server's filesystem. You must
# explicitly permit access to web content directories in other
# <Directory> blocks below.
#
<Directory />
    AllowOverride none
#    Require all denied
```

```
26      Require all granted
27  </Directory>
28
29  DocumentRoot "/var/www/html"
30
31  #
32  # Relax access to content within /var/www.
33  #
34  <Directory "/var/www">
35      AllowOverride None
36      # Allow open access:
37      Require all granted
38  </Directory>
39
40  # Further relax access to the default document root:
41  <Directory "/var/www/html">
42      Options Indexes FollowSymLinks
43
44      AllowOverride None
45
46      Require all granted
47  </Directory>
48
49  #
50  # DirectoryIndex: sets the file that Apache will serve if a directory
51  # is requested.
52  #
53  <IfModule dir_module>
54      DirectoryIndex index.html
55  </IfModule>
56
57  #
58  # The following lines prevent .htaccess and .htpasswd files from being
59  # viewed by Web clients.
60  #
61  <Files ".ht*">
62      Require all denied
63  </Files>
64
65  ErrorLog "logs/error_log"
66
67  LogLevel warn
68
69  <IfModule log_config_module>
70      #
71      # The following directives define some format nicknames for use with
72      # a CustomLog directive (see below).
73      #
74      LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
75      LogFormat "%h %l %u %t \"%r\" %>s %b" common
76
77      <IfModule logio_module>
78        # You need to enable mod_logio.c to use %I and %O
79        LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O"
      combinedio
80      </IfModule>
81
82      #
83      # The location and format of the access logfile (Common Logfile Format).
84      # If you do not define any access logfiles within a <VirtualHost>
85      # container, they will be logged here.  Contrariwise, if you *do*
86      # define per-<VirtualHost> access logfiles, transactions will be
87      # logged therein and *not* in this file.
88      #
89      #CustomLog "logs/access_log" common
90
91      #
92      # If you prefer a logfile with access, agent, and referer information
93      # (Combined Logfile Format) you can use the following directive.
94      #
95      CustomLog "logs/access_log" combined
96  </IfModule>
97
98  <IfModule alias_module>
99      #
```

```
100     # Redirect: Allows you to tell clients about documents that used to
101     # exist in your server's namespace, but do not anymore. The client
102     # will make a new request for the document at its new location.
103     # Example:
104     # Redirect permanent /foo http://www.example.com/bar
105
106     #
107     # Alias: Maps web paths into filesystem paths and is used to
108     # access content that does not live under the DocumentRoot.
109     # Example:
110     # Alias /webpath /full/filesystem/path
111     #
112     # If you include a trailing / on /webpath then the server will
113     # require it to be present in the URL.  You will also likely
114     # need to provide a <Directory> section to allow access to
115     # the filesystem path.
116
117     #
118     # ScriptAlias: This controls which directories contain server scripts.
119     # ScriptAliases are essentially the same as Aliases, except that
120     # documents in the target directory are treated as applications and
121     # run by the server when requested rather than as documents sent to the
122     # client.  The same rules about trailing "/" apply to ScriptAlias
123     # directives as to Alias.
124     #
125 ScriptAlias /cgi-bin/ /var/www/cgi-bin/
126
127 </IfModule>
128
129 #
130 # "/var/www/cgi-bin" should be changed to whatever your ScriptAliased
131 # CGI directory exists, if you have that configured.
132 #
133 <Directory "/var/www/cgi-bin">
134     AllowOverride None
135     Options None
136     Require all granted
137 </Directory>
138
139 <IfModule mime_module>
140
141     TypesConfig /etc/mime.types
142
143     #AddType application/x-gzip .tgz
144
145     #AddEncoding x-compress .Z
146     #AddEncoding x-gzip .gz .tgz
147
148     AddType application/x-compress .Z
149     AddType application/x-gzip .gz .tgz
150
151     #AddHandler cgi-script .cgi
152
153     # For type maps (negotiated resources):
154     #AddHandler type-map var
155
156     AddType text/html .shtml
157     AddOutputFilter INCLUDES .shtml
158 </IfModule>
159
160 AddDefaultCharset UTF-8
161
162 <IfModule mime_magic_module>
163     #
164     # The mod_mime_magic module allows the server to use various hints from the
165     # contents of the file itself to determine its type.  The MIMEMagicFile
166     # directive tells the module where the hint definitions are located.
167     #
168     MIMEMagicFile conf/magic
169 </IfModule>
170
171 #EnableMMAP off
172 EnableSendfile on
173
174 # Supplemental configuration
```

```
175 #
176 # Load config files in the "/etc/httpd/conf.d" directory, if any.
177 IncludeOptional conf.d/*.conf
178 ServerTokens Minor
```

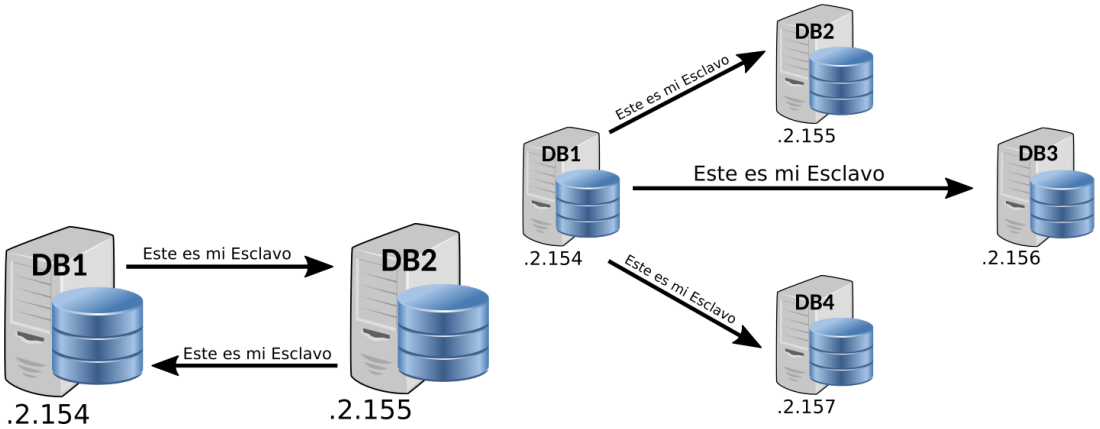
httpd.cfg

8.5. Configuración especifica de MariaDB

Por ultimo para poder dar mas fiabilidad a nuestra granja web hemos utilizado una configuración de base de datos en modo Maestro-Maestro, esto quiere decir que da igual en el servidor que modifiquemos los datos puesto que se va a guardar la información en ambos, así de esta forma conseguiremos un nivel mas de seguridad y por supuesto también de alta disponibilidad, ya que si por algún casual uno de los servidores se estropeara el otro podría dar servicio sin problemas a nuestra granja de 2 servidores web. Claro esta en que esta configuración esta dimensionada para un granja de 4 servidores web como mucho, es por eso que si tuviéramos que, dar servicio a un mayor numero de servidores frontend, tendríamos que pensar en utilizar, una configuración distinta.

Por falta de tiempo para realizar las pruebas necesarias, no he podido montar esta configuración pero, si me gustaría dejarla explicada para futuras pruebas. Bien, hasta el momento tal y como tenemos montada nuestra base de datos bastaría con incluir otro servidor y añadirlo como esclavo a cualquiera de los 2 servidores actuales, y luego añadir como esclavo este ultimo servidor al nuevo, de esta manera tendriamos una configuración maestro-maestro entre grupos de servidores y todos ellos serian maestros.

El problema de esta configuración, es que si empezamos a añadir servidores de esta manera, cuando agreguemos información al primer servidor, habrá una latencia bastante alta en el supuesto que tengamos 10 servidores, por eso, creo que la configuración ideal sería utilizar una configuración en estrella, es decir, si añadiéramos un servidor nuevo a nuestra base de datos, lo configuraríamos de manera que los dos servidores que hay actualmente en funcionamiento serian esclavos de este nuevo servidor, y a su vez en los dos servidores que ya están en funcionamiento le añadiríamos un esclavo que seria nuestro nuevo servidor, así de esta forma tendríamos una configuración en estrella tal y como se muestra en las siguientes figuras (8.1b).



(a) Configuración antes de Estrella (b) Configuración después de Estrella

Figura 8.1: Hardware y software de jetNexus

Por supuesto esta operación la tendríamos que hacer en todos los servidores de nuestra base de datos así de esa forma nuestros servidores frontend, harían las peticiones a un solo servidor que haría de balanceador de carga para nuestra granja de servidores de base de datos y reduciríamos las latencias de replicación.

Comandos usuados para la configuración

```
#!/bin/bash
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="create user 'replicauser'@'%' identified by 'contraseña'"
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="grant replication slave on *.* to 'replicauser'@'%'"
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="flush privileges"
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="flush tables"
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="flush tables with read lock"
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="show master status"
+--------------------+----------+--------------+------------------+
| File               | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+--------------------+----------+--------------+------------------+
| mariadb-bin.000001 |      245 |              |                  |
+--------------------+----------+--------------+------------------+
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="create user 'replicauser'@'%' identified by 'contraseña'"
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="grant replication slave on *.* to 'replicauser'@'%'"
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="flush privileges"
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="flush tables"
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="flush tables with read lock"
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="show master status"
+--------------------+----------+--------------+------------------+
| File               | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+--------------------+----------+--------------+------------------+
| mariadb-bin.000002 |      245 |              |                  |
+--------------------+----------+--------------+------------------+
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="stop slave"
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="change master to master_host='IP_DE_DB1', MASTER_USER='replicauser', MASTER_PASSWORD='contraseña', MASTER_LOG_FILE='mariadb-bin.000001', MASTER_LOG_POS=245"
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="start slave"
[usuario@DB2 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="unlock tables"
echo .
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="stop slave"
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="change master to master_host='IP_DE_DB2', MASTER_USER='replicauser', MASTER_PASSWORD='contraseña', MASTER_LOG_FILE='mariadb-bin.000002', MASTER_LOG_POS=245"
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="start slave"
[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="unlock tables"

[usuario@DB1 /]# mysql --host=localhost --user=root --password=contraseña --database="contactos" --execute="show slave status\G"

Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.50.159
                  Master_User: replicauser
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mariadb-bin.000004
          Read_Master_Log_Pos: 245
               Relay_Log_File: relay-bin.000007
                Relay_Log_Pos: 531
        Relay_Master_Log_File: mariadb-bin.000004
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: contactos
```

```
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 245
              Relay_Log_Space: 1105
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 1
```

comandos.sh

Configuración de *mysql*

```
# bind-address            = 127.0.0.1
server-id                 = 1
report_host               = 192.168.2.154
log_bin                   = /var/log/mariadb/mariadb-bin
log_bin_index             = /var/log/mariadb/mariadb-bin.index
relay_log                 = /var/log/mariadb/relay-bin
relay_log_index           = /var/log/mariadb/relay-bin.index
sync_binlog               = 1
#replicate-do-db          = contactos

datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock

# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0

# Settings user and group are ignored when systemd is used.
# If you need to run mysqld under a different user or group,
# customize your systemd unit file for mariadb according to the
# instructions in http://fedoraproject.org/wiki/Systemd

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

#
# include all files from the config directory
#
!includedir /etc/my.cnf.d
```

my.cnf

9. Conclusiones

En este laboratorio he pretendido realizar una instalación lo mas parecida posible a un entorno real, por supuesto hay cosas que por falta de tiempo y de recursos no he podido implementar como es el caso de un servidor de correo IMAP replicado, o por ejemplo un servidor de FTP con almacenamiento compartido por los servidores web, de tal forma que fuese una posible opción a utilizar por los usuarios a la hora de

albergar sus propias paginas web, aunque es un protocolo que esta en decadencia todavía se sigue usando.

Sin embargo si he podido contemplar las opciones de Alta Disponibilidad gracias a la configuración de maestro-maestro de las bases de datos y el balanceador de carga para repartir el trabajo entre los servidores web, otra cosa a destacar en el laboratorio, es que las copias de seguridad son posibles con un mínimo de esfuerzo, puesto que los comandos para las mismas están puestos en tareas programadas de CRON a falta de ser activadas.

Quizás una de las cosas que me hubiera gustado implementar pero no me ha sido posible (que conste que lo he intentado) por falta de recursos, hubiera sido crear el dominio empresarial (midominio.test) en un entorno de Active Directory de Microsoft y unir toda esta infraestructura a la configurada en Linux, para ver que tal se comportaban los sistemas.

Referencias

[1] Definición de servidor dns, Consultado el 16 de mayo de 2015. http://es.wikipedia.org/wiki/Domain_Name_System.

[2] Netscaler vpx prices, Consultado el 16 de mayo de 2015. http://store.citrix.com/store?SiteID=citrix&Action=DisplayProductDetailsPage&productID=315172500.

[3] Portal español sobre postgresql, Consultado el 16 de mayo de 2015. http://www.postgresql.org/.

[4] Portal español sobre postgresql, Consultado el 16 de mayo de 2015. http://www.postgresql.org.es/.

[5] Página oficial de apache server, Consultado el 16 de mayo de 2015. http://httpd.apache.org/.

[6] Página oficial de ipcop firewall, Consultado el 16 de mayo de 2015. http://www.ipcop.org/.

[7] Página oficial de jetnexus, Consultado el 16 de mayo de 2015. http://www.jetnexus.com/load-balancer/tech-specs/.

[8] Página oficial de mariadb, Consultado el 16 de mayo de 2015. https://mariadb.org/.

[9] Página oficial de webmin, Consultado el 16 de mayo de 2015. http://www.webmin.com/.

[10] Rambler, portal de búsqueda, Consultado el 16 de mayo de 2015. http://es.wikipedia.org/wiki/Rambler_(porta).

[11] The reliable, high performance tcp/http load balancer, Consultado el 16 de mayo de 2015. http://www.haproxy.org.

[12] Software para configurar servidor dns, Consultado el 16 de mayo de 2015. https://www.isc.org/downloads/bind/.

[13] Using nginx as http load balancer, Consultado el 16 de mayo de 2015. http://nginx.org/en/docs/http/load_balancing.html.

[14] White paper firewall cisco asa 5500 series, Consultado el 16 de mayo de 2015. http://www.cisco.com/c/en/us/products/collateral/security/asa-5500-series-next-generation-firewalls/datasheet-c78-733916.html.

[15] Wiki español sobre postgresql, Consultado el 16 de mayo de 2015. https://wiki.postgresql.org/wiki/Espa~nol/.

[16] Wikipedia mariadb, Consultado el 16 de mayo de 2015. http://es.wikipedia.org/wiki/MariaDB.