UNIVERSIDAD LUTERANA SALVADOREÑA
FACULTAD DE CIENCIAS DEL HOMBRE Y LA NATURALEZA

Materia: Sistemas Operativos de Redes.

catedrático: Ingeniero Manuel Flores.

Tema: Perfil de proyecto, “Cluster web con balanceo de Carga y alta disponibilidad”

fecha de entrega: 31/Octubre/2015

Estudiantes:

Carnet	**Apellidos**	**Nombres**	**Participación**
DP02110390	Diaz Palacios	Rafael Antonio	(de 100%-100%)
ME01121555	Mazana Escobar	Manuel Antonio	(de 100%-100%)

Índice de contenido

Índice de figuras

Índice de tablas

Introducción

El presente documento contiene el perfil de nuestro proyecto "***Cluster web con balanceo de carga y alta disponibilidad***" a realizarse en la materia de Sistemas Operativos de redes. Dentro del documento, se presentan las diferentes tecnologías asociadas, una descripción de lo que se pretende implementar así como un estudio básico de factibilidades.

Un cluster es un conjunto de computadoras que trabajan de manera colectiva,compartiendo recursos entre sí, para operar como una sola máquina.

¿Cual es el propósito de tener un cluster web?

Como sabemos hoy en día los sitios web son parte fundamental de las empresas, en el área de marketing para dirigir publicidad, promocionar servicios, entre otras cosas. también se utilizan para almacenar información de consulta constante para los empleados. muchas personas en todo el mundo pueden estar realizando solicitudes a un servidor para poder visualizar una misma página web, con tantas peticiones es común que el tiempo de respuesta en algún momento del día sea prolongado, incluso puede que la página esté inaccesible por una cierta cantidad de tiempo. Es por eso que trataremos de resolver este inconveniente en nuestro servidor para que así esté disponible en linea la mayor parte del tiempo posible y que el usuario final pase desapercibido casi cualquier falla presentada en el servidor, además utilizando las bondades de los sistemas de multiprocesamiento evitaremos cuellos de botella en las peticiones al servidor, usando para ello un balanceo de carga, logrando así gestionar las solicitudes de usuarios al servidor,con posibilidad a ser un sistema escalable en cuanto se requiera.

Objetivos

Objetivos Generales

- Desarrollar un cluster web de alta disponibilidad con balanceo de carga, con fines autodidacticos, a realizarse en la feria de tecnologías FLISOL , Universidad Luterana Salvadoreña en el ciclo II -2015.

Objetivos Específicos

- Conocer información relevante sobre conceptos de balanceador de carga, clustering web y alta disponibilidad.
- Mostrar paso a paso los comandos y las herramientas a utilizar para el montaje del proyecto.
- Ejecutar el desarrollo del proyecto, para darlo a conocer en la feria FLISOL 2015.

Marco teórico.

Clusters en informática.

Definición de Clustering.

El origen del término Cluster y del uso de este tipo de tecnología es desconocido pero se puede considerar que comenzó a finales de los años cincuenta y principios de los sesenta.

cluster se le denomina a los conjuntos de computadoras construidos mediante la utilización de componentes de hardware comunes y que se comportan como si fuesen una única computadora. Hoy en día desempeñan un papel importante en la solución de problemas de las ciencias, las ingenierías y del comercio moderno. Inventado posiblemente por Gene Amdahl de IBM, que en 1967 publicó lo que ha llegado a ser considerado como el inicio del procesamiento paralelo. El cómputo con clusters surge como resultado de la convergencia de varias tendencias actuales que incluyen la disponibilidad de microprocesadores económicos de alto rendimiento y redes de alta velocidad, el desarrollo de herramientas de software para cómputo distribuido de alto rendimiento, así como la creciente necesidad de potencia computacional para aplicaciones que la requieran.

Los clusters son usualmente empleados para mejorar el rendimiento y/o la disponibilidad por encima de la que es provista por un solo computador típicamente siendo más económico que computadores individuales de rapidez y disponibilidad comparables.

De un cluster se espera que presente combinaciones de los siguientes servicios:

1. Alto rendimiento
2. Alta disponibilidad
3. Balanceo de carga
4. Escalabilidad

Clasificación de los Cluster, según su configuración.

- **Homogéneos:** Pueden tener todos la misma configuración de hardware y sistema operativo.
- **Semi homogéneos**: Diferente rendimiento pero con arquitecturas y sistemas operativos similares.
- **Heterogéneos:** Tienen diferente hardware y sistema operativo, lo que hace más fácil y económica su construcción.

Para que un clúster funcione como tal, no basta solo con conectar entre sí los ordenadores, sino que es necesario proveer un sistema de manejo del clúster, el cual se encargue de interactuar con el usuario y los procesos que corren en él para optimizar el funcionamiento.

Ilustración 1: Ejemplo de Beowulf Cluster.por Alex Schenck.

Soluciones libres para Clustering.

Como mencionamos anteriormente para realizar Clustering es necesario proveer un sistema que se encargue de manejar las operaciones entre los equipos ya sean físicos o virtuales.

a continuación listamos algunos de los investigados.

- **Heartbeat**

Heartbeat diseñado para detectar cuando se ha caído un servicio en una máquina y administración de cluster.

Heartbeat puede utilizar dispositivos ethernet y de serie para comunicarse con los otros nodos del cluster.

- **Kimberlite**

Kimberlite es una solución de cluster para clusters de dos nodos y permite la conexión a un disco compartido de quorum vía SCSI o por fibra (SAN).

- **Ultramonkey**

Ultramonkey es una solución para balanceo de carga y alta disponibilidad basada en LVS.

Ultramonkey utiliza:

1. Heartbeat en los balanceadores para detectar cuando ha caído un servicio, monitorizar los servidores a balancear mediante ldirectord y controlar la IP del servicio balanceado.

2. ldirectord en los balanceadores para realizar el health-checking de los servidores balanceados.

- **Red Hat Cluster Suite**

Red Hat Cluster Suite es la solución de Red Hat para clustering y alta disponibilidad.

Esta solución consta de:

1. GFS sistema de ficheros de acceso concurrente.
2. CLVM o Clustered LVM.
3. Piranha interface gráfico para LVS.
4. system-config-cluster herramienta gráfica para la configuración del cluster.
5. ccsd demonio que hace accesible la configuración del cluster por otros nodos.
6. cman demonio para la administración del cluster.
7. qdiskd demonio para el manejo de discos de quorum..
8. fenced demonio para el fencing de dispositivos.
9. Rgmanager.

Alta disponibilidad.

¿Que es la alta disponibilidad?

La disponibilidad es una de las características de las arquitecturas empresariales que mide el grado con el que los recursos del sistema están disponibles para su uso por el usuario final a lo largo de un tiempo dado. Ésta no sólo se relaciona con la prevención de caídas del sistema (también llamadas tiempos fuera de línea, downtime u offline), sino incluso con la percepción de "caída" desde el punto de vista del usuario: cualquier circunstancia que nos impida trabajar productivamente con el sistema – desde tiempos de respuesta prolongados, escasa asistencia técnica o falta de estaciones de trabajo disponibles – es considerada como un factor de baja disponibilidad.

¿Cómo medimos la disponibilidad?

De primera instancia, todo sistema debe tener establecido un Acuerdo de Nivel de Servicio (Service Level Agreement – SLA) que defina cuánto tiempo y en qué horarios debe estar en línea. En el caso de aplicaciones de baja criticidad, dicho SLA puede ser de 8×5 horas a la semana excluyendo días festivos; para sistemas con mayor criticidad como una red de cajeros automáticos se tienen niveles de servicio que alcanzan las 24 horas al día, los 365 días del año. Así entonces, suponiendo un sistema con un SLA de 24×365 podríamos calcular su disponibilidad de la siguiente manera:

Disponibilidad = ((A – B)/A) x 100 por ciento)

Donde:
***A** = Horas comprometidas de disponibilidad: 24 x 365 = 8,760 Horas/año.*
***B** = Número de horas fuera de línea (Horas de "caída del sistema" durante el tiempo de disponibilidad comprometido). Por ejemplo: 15 horas por falla en un disco; 9 horas por mantenimiento preventivo no planeado.*

así entonces:
Disponibilidad = ((8,760 – 24)/8,760) x 100 por ciento) = 99.726%
Cuando se realicen negociaciones para definir objetivos de disponibilidad con los usuarios, es necesario hacerlos conscientes de las implicaciones técnicas y económicas, como se muestra en la siguiente tabla:

Disponibilidad (%)	**Tiempo offline/año**	**Tiempo offline/mes**	**Tiempo offline/día**
90%	36.5 días	73 hrs	2.4 hrs
95%	18.3 días	36.5 hrs	1.2 hrs
98%	7.3 días	14.6 hrs	28.8 min
99%	3.7 días	7.3 hrs	14.4 min
99.5%	1.8 días	3.66 hrs	7.22 min
99.9%	8.8 hrs	43.8 min	1.46 min
99.95%	4.4 hrs	21.9 min	43.8 s
99.99%	52.6 min	4.4 min	8.6 s
99.999%	5.26 min	26.3 s	0.86 s
99.9999%	31.5 s	2.62 s	0.08 s

Tabla 1: Disponibilidad para un sistema 24×7 y tiempos de caída permitidos.

Estos números (especialmente aquellos que pasan de la marca del 99.5% de disponibilidad) son difíciles de alcanzar, ya que es necesario poder recuperarse ante caídas del sistema de manera transparente. La capacidad e intervalo de tiempo necesarios para recuperarse ante tal eventualidad son directamente dependientes de:

- La complejidad del sistema.
- La severidad del problema.
- La disponibilidad del personal de soporte.
- La madurez en materia de administración del sistema y sus operaciones.
- Otros factores técnicos o de gestión: falta de refacciones, fallas en la cadena de escalamiento, etc.

Balanceo de carga

Definición de balanceo de carga

El balance o balanceo de carga es un concepto usado en informática que se refiere a la técnica usada para compartir el trabajo a realizar entre varios procesos, ordenadores, discos u otros recursos. Está íntimamente ligado a los sistemas de multiprocesamiento, o que hacen uso de más de una unidad de procesamiento para realizar labores útiles.

Para nuestro proyecto hablaremos puntualmente de Balanceo de carga para un servidor web. El balanceo de carga para un web server se mantiene gracias a un algoritmo que divide de la manera más equitativa posible el trabajo, para evitar los así denominados cuellos de botella.

Hay balanceadores de carga tipo round-robin (uno a uno) y por pesos (que son capaces de saber cuál de los nodos está más libre y lanzarle la petición). El más conocido es LVS, sin embargo hay otros, como el Red Hat Piranha.

Y en la plataforma para Windows Server se tiene al ISA Server (Microsoft Internet Security and Acceleration Server).

Web server

Definición de web server

Básicamente, un servidor web sirve contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante el protocolo de HTTP.Se pueden utilizar varias tecnologías en el servidor para aumentar su potencia más allá de su capacidad de entregar páginas HTML; éstas incluyen scripts CGI, seguridad SSL y páginas activas del servidor (ASP).

Apache

Apache es un poderoso servidor web, cuyo nombre proviene de la frase inglesa “a patchy server” y es completamente libre, ya que es un software Open Source y con licencia GPL. Una de las ventajas más grandes de Apache, es que es un servidor web multiplataforma, es decir, puede trabajar con diferentes sistemas operativos y mantener su excelente rendimiento.

Desde el año 1996, es el servidor web más popular del mundo, debido a su estabilidad y seguridad. Apache sigue siendo desarrollado por la comunidad de usuarios desarrolladores que trabaja bajo la tutela de Apache Software Foundation.

Principales características de Apache

Entre las principales características de Apache, se encuentran las siguientes:

- Soporte de seguridad SSL y TLS.
- Puede realizar autentificación de datos utilizando SGDB.
- Puede dar soporte a diferentes lenguajes, como Perl, PHP, Python y tcl.

Características

- Apache es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos HTTP.
- Multiplataforma.
- Modular: Puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la API de programación de módulos, para el desarrollo de módulos específicos.
- Extensible: gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor.

Ilustración 2: Logo de Apache software Foundation | Apache License, Version 2.0

Descripción del proyecto final.

La idea de nuestro proyecto es configurar un clúster de servidores web Apache de dos nodos que proporcionen alta disponibilidad. Para ellos tendremos dos servidores apache conectado a un balanceador de carga para que divida las solicitudes entrantes entre los nodos de apache. Sabemos que un balanceador de carga para los dos nodos de apache puede llegar a ser otro punto de falla, pues si este se cae el balanceador de carga las solicitudes no se repartirán a los nodos del servidor Apache. Como queremos proporcionar alta disponibilidad, El balanceador de carga, también se compondrá de dos nodos así si un Balanceador de carga falla, el otro responderá logrando mayor disponibilidad en el servidor.

La ventaja de utilizar un equilibrador de carga en comparación con el uso de DNS round robin según lo investigado es que el balanceador tratara de enviar las solicitudes al nodo con menos carga. Además de eso, si uno de los nodos de Apache se cae, el balanceador de carga se dará cuenta de eso y dirigirá todas las peticiones entrantes al nodo restante .Para esta configuración, necesitamos cuatro nodos (dos nodos Apache y dos nodos equilibrador de carga) y cinco direcciones IP: una para cada nodo y una dirección IP virtual que será compartida por los nodos del equilibrador de carga y se utiliza para las solicitudes HTTP entrantes. Utilizaremos Ultramonkey para el balanceo de carga.

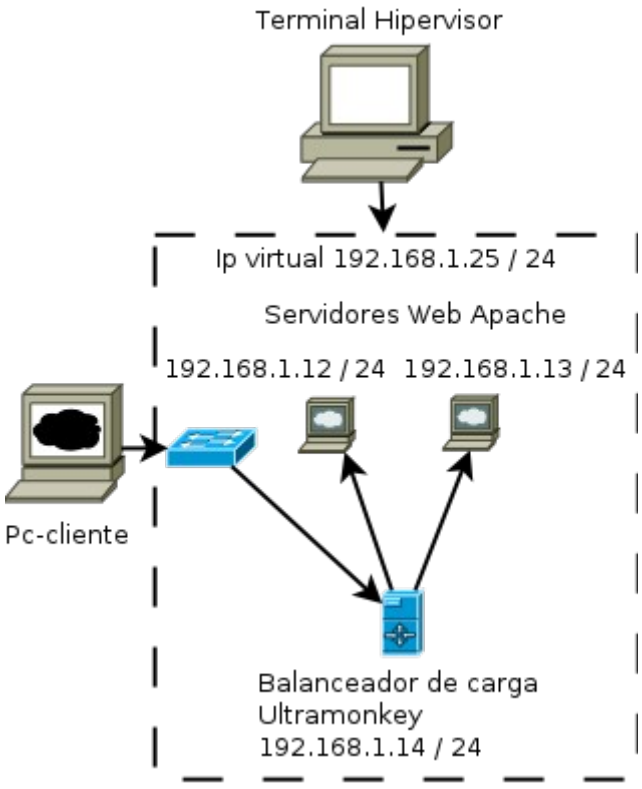


Ilustración 3: Diagrama de red del montaje del proyecto

Diagrama de Gantt

Actividades	Agosto				Septiembre				Octubre				Noviembre				Fecha finalización
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	
Investigación																	
Lectura de Clusters																	15/08/15
Lectura de balanceo de carga																	18/08/15
Lectura de Alta disponibilidad																	19/08/15
Elaboración del perfil del proyecto																	20/08/15
Lectura y practica de LVS																	28/08/15
Lectura y practica de Heartbeat																	05/09/15
Lectura y practica de Ultramonkey																	08/09/15
Instalación de maquinas virtuales.																	15/09/15
Instalación y configuración del balanceador de carga																	26/09/15
Instalación y configuración del los servidores apache																	03/10/15
Instalación y configuración de los clusters																	10/10/15
Pruebas y depuración																	31/10/15
Documentación final																	07/11/15
Preparación de materiales para exposición																	14/11/15

Viabilidad

El Proyecto reúne características, condiciones técnicas y operativas que aseguran el cumplimiento de sus metas y objetivos. recoge las experiencias de estudiantes de la carrera de licenciatura en ciencias de la computación que trabajan en su ejecución.

El Proyecto y su estructura responde a una estrategia que busca generar material didáctico pero a la vez busca generar alternativas tecnológicas para distintos sectores empresariales que puedan hacer uso de estas herramientas, o desconocen este tipo de soluciones que claramente pueden aportar a la economía de sus empresas y a la productividad de las mismas.

La Tecnología a utilizar en el proyecto es Libre, lo que implica ahorros en costos de licenciamiento y evita restricciones que las herramientas privativas imponen. Además al ser herramientas libres aseguramos que el conocimiento se pueda compartir y seguir desarrollando quizá por otras personas interesadas en ahondar mas en el tema.

Presupuesto.

ACTIVIDADES	INSUMOS	VALOR UNITARIO	SUB-TOTAL
RECURSOS HUMANOS	Gastos de asesoría 2 Estudiantes 1 Docente Asesor	$30 por asesoría N° de asesorías = 6	$180.00
TOTAL DEL RUBRO A			$180.00
RECURSOS MATERIALES	2 computadoras Core 2 duo de 3 GHZ 4 GB de Ram y tarjeta de vídeo de 1 GB.	$700	$1400.00
	Impresión del documento	$0.30 (35)	$10.05
TOTAL DEL RUBRO B			$1410.05
TOTAL			**$1590.05**

Desarrollo del prototipo.

Instalación de las maquinas virtuales

Herramientas necesarias

- ISO de instalación del sistema operativo Debian Jessi 8.0
- Apache2
- Hearbeat
- Virtualbox

Procedimiento de Instalación

Instalación de virtual box.

En consola como root actualizamos los repositorios y los paquetes, e instalamos el paquete virtualbox y también virtualbox-guest-x11:

#su(contraseña)

#aptitude update

#aptitude upgrade

#aptitude install virtualbox virtualbox-guest-x11

Automaticamente instala las dependencias : dkms,virtualbox-dkms y virtualbox-qt.

Agregar nuestro usuario al grupo vboxuser

Tenemos que agregar nuestro usuario al grupo vboxusers para poder utilizar los

Dispositivos USB en las maquinas virtuales que vayamos a instalar. Para hacer esto,

debemos abrir la terminal y ejecutar el siguiente comando como superusuario:

#adduser tu_usuario vboxusers

Para comprobar que el usuario se agrego correctamente al grupo, ejecutamos

#adduser tu_usuario vboxusers

Y nos dirá “El usuario “nombre” ya es un miembro de 'vboxusers'.

Después reiniciamos el sistema.

Creación de las maquinas virtuales:

Una vez instalado Virtualbox

Vamos a montar 2 maquinas virtuales para apache y una maquina virtual para el balanceador de carga.

Instalación del sistema operativo en las maquinas virtuales

Para la instalación del sistema operativo en las maquinas virtuales utilizaremos la siguiente configuración

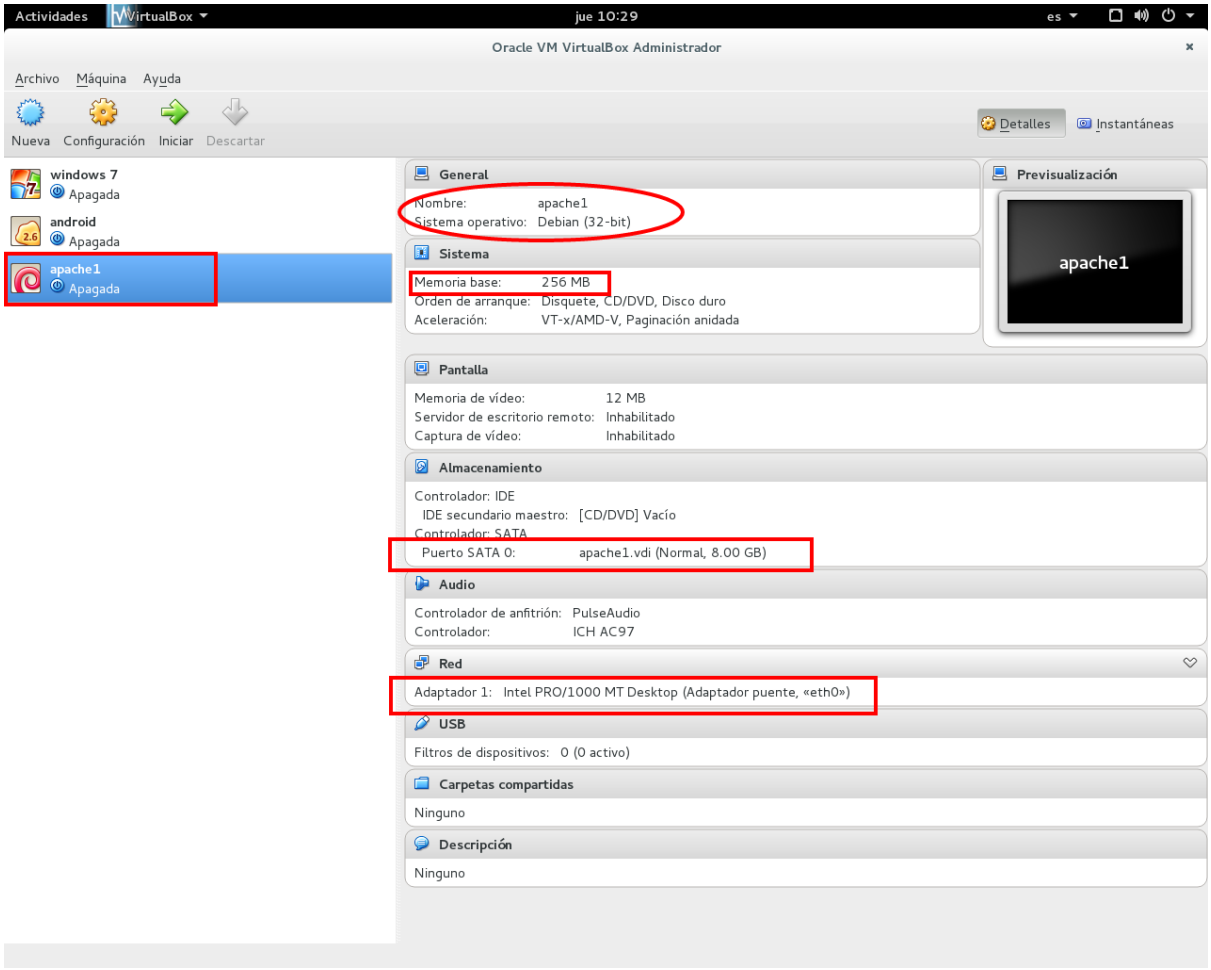


Ilustración 4: Parámetros de instalación, para las maquinas virtuales.

- Sistema operativo: Debian GNU/Linux de 32 bits
- Memoria del sistema: 256 mb
- Disco duro: 8GB
- Adaptador de red: modo puente.

No es necesario hacer la instalación de las otras maquinas virtuales, simplemente se instala una maquina con los recursos mencionados y clonamos el resto.

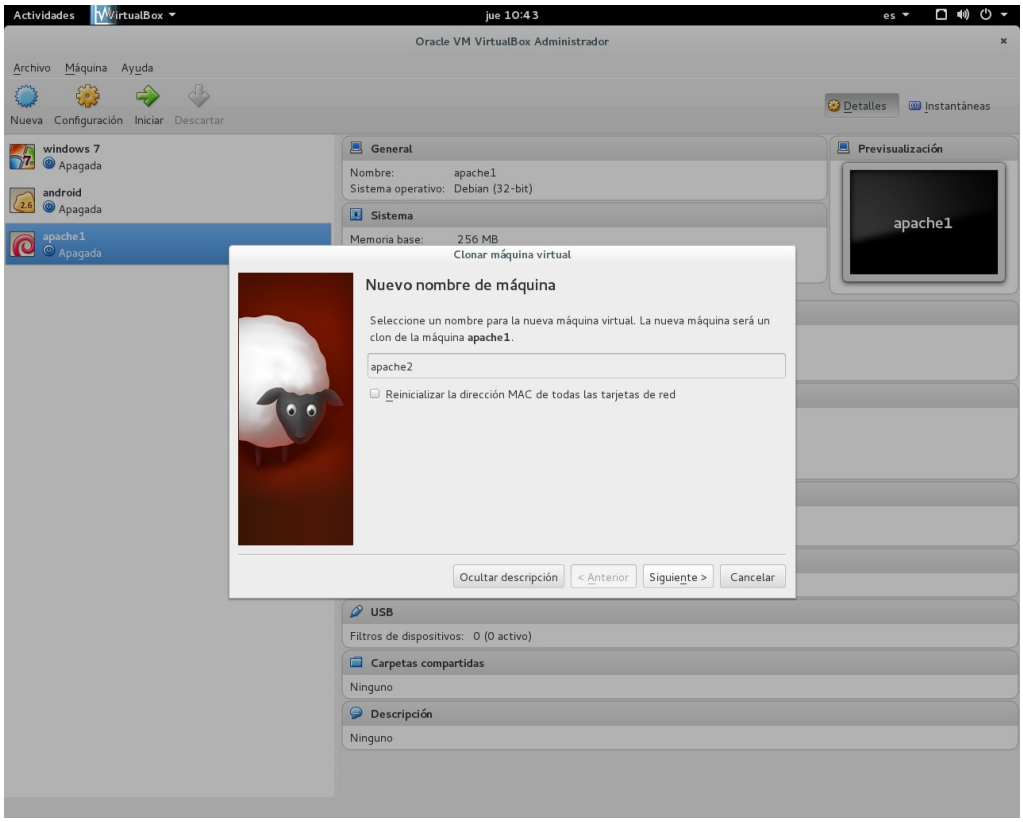


Ilustración 5: Asistente de virtualbox para clonar maquinas virtuales

Para clonar una maquina virtual hacemos click derecho sobre la maquina en el administrador de virtualbox, seleccionando la opción de clonar y el asistente de virtualbox nos pedirá el nombre que le daremos a la nueva maquina virtual.

Para el ejemplo usaremos las ip siguientes

•Apache1:(webserver1) - IP address: 192.168.15.180; Apache document root: /var/www

•Apache2: (webserver2) - IP address: 192.168.15.181; Apache document root: /var/www

•Dirección Virtual IP: 192.168.15.182(se usara para las peticiones entrantes)

Instalación de Heratbeat para el clustering de servidores apache y la alta disponibilidad

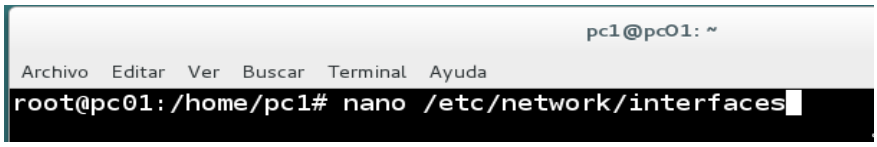
Primero configuramos nuestras IP estáticas en cada maquina.

Recomendación:

poner las IP en forma ascendente ya que HEARTBEAT reconoce la IP de la maquina principal de esta misma forma:

configuración de interfaces en el archivo **/etc/network/interface** en cada maquina

 # nano /etc/network/interface



servicio web principal:

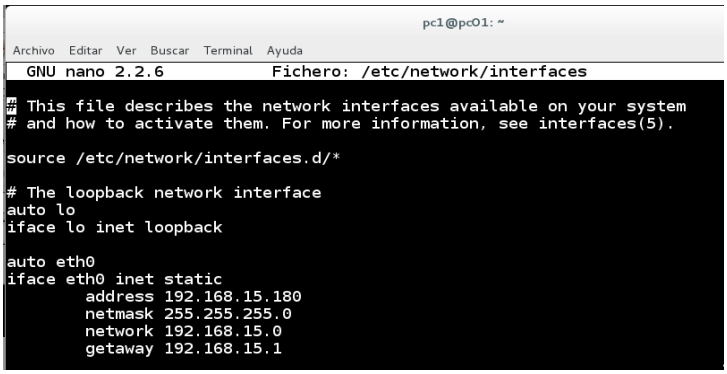
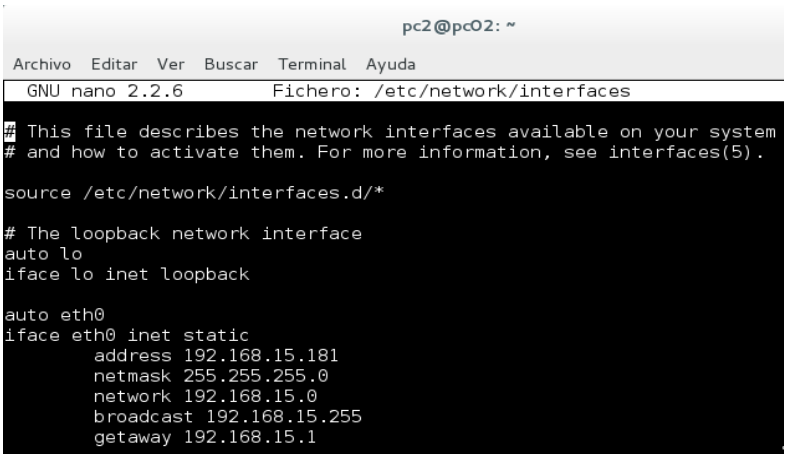


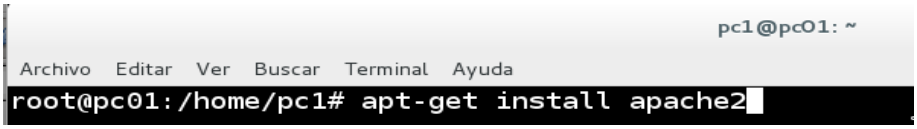
Ilustración 6: Configuraciones del archivo que controla las interfaces de red

servicio web secundario:



primero instalamos el servidor web apache:

apt-get install apache2



Luego detenemos el servicio

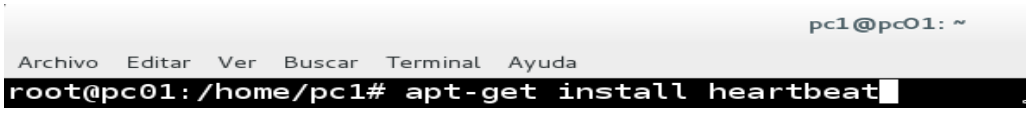
service apache2 stop

Para evitar que el servicio apache2 se levante al inicio

update-rc.d -f apache2 disable

A continuación instalamos HEARTBEAT:

apt-get install heartbeat



Tras la instalación de HEARTBEAT habrá aparecido en la ruta **/usr/share/doc/heartbeat**, la cual contiene 3 ficheros que debemos configurar para que HEARTBEAT funcione.

Estos 3 ficheros son:

authkeys: Fichero de autenticación entre nodos.

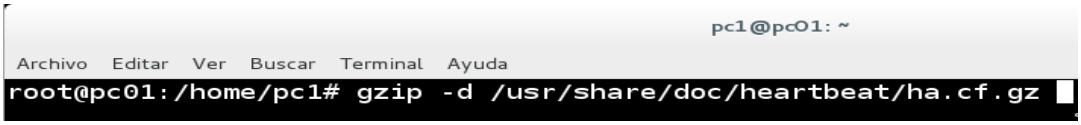
ha.cf: Fichero principal de configuración de HEARTBEAT.

Haresources: Fichero donde se indica cuál sera el nodo activo. Asi como los servicios a gestionar.

Luego de haber instalado HEARTBEAT descomprimimos y copiamos los ficheros de configuración de HEARTBEAT en la ruta /etc/ha.d

Descomprimimos **ha.cf.gz**

#gzip -d /usr/share/doc/heartbeat/ha.cf.gz



Descomprimimos **haresources.gz**

#gzip -d /usr/share/doc/heartbeat/haresources.gz

Ahora copiamos el fichero **authkeys**

#cp /usr/share/doc/heartbeat/authkay /etc/ha.d/

Ahora copiamos el fichero **ha.cf**

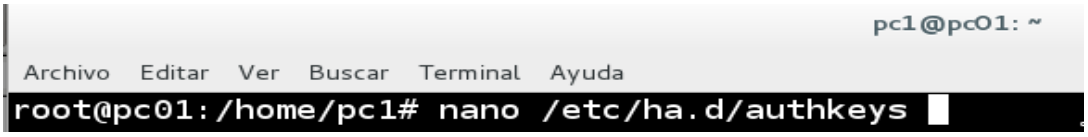
#cp /usr/share/doc/heartbeat/ha.cf /etc/ha.d/

Ahora copiamos el fichero **haresource**

#cp /usr/share/doc/heartbeat/haresource /etc/ha.d/

authkeys: en este fichero estableceremos la contraseña de autenticación compartida por los dos nodos.

#nano /etc/ha.d/authkeys



```
#       sha1 is believed to be the "best", md5 next best.
#
#       crc adds no security, except from packet corruption.
#               Use only on physically secure networks.
#
auth 1
1 crc
#2 sha1 HI!
#3 md5 Hello!
```

Ilustración 7: Fichero Authkeys

Le damos permisos para que solo pueda ser accedido por el propietario del mismo

#chmod 600 /etc/ha.d/authkeys

esta configuración sera igual en los dos nodos

ha.cf: este fichero principal de configuración de HEARTBEAT. Aquí descomentaremos algunas lineas que son esenciales para el correcto funcionamiento de HEARTBEAT.

#nano /etc/ha.d/ha.cf

#Tiempo transcurrido entre el envío de cada uno de los latidos (segundos)

keepalive 2

#Tiempo transcurrido hasta declarar al nodo como caído (segundos)

deadtime 10

```
#
#
#       keepalive: how long between heartbeats?
#
keepalive 2
#
#       deadtime: how long-to-declare-host-dead?
#
#               If you set this too low you will get the problematic
#               split-brain (or cluster partition) problem.
#               See the FAQ for how to use warntime to tune deadtime.
#
deadtime 10
#
```

#Tiempo transcurrido para comenzar a levantar los servicios (segundos)

initdead 120

#Puerto para la comunicación UDP

udpport 694

```
#
#       Very first dead time (initdead)
#
#       On some machines/OSes, etc. the network takes a while to come up
#       and start working right after you've been rebooted.  As a result
#       we have a separate dead time for when things first come up.
#       It should be at least twice the normal dead time.
#
initdead 120
#
#
#       What UDP port to use for bcast/ucast communication?
#
udpport 694
```

esta configuración sera igual en los dos nodos

#Interfaz utilizado para enviar los mensajes de broadcast

bcast eth0

```
#
#
#       What interfaces to broadcast heartbeats over?
#
bcast   eth0            # Linux
#bcast  eth1 eth2       # Linux
#bcast  le0             # Solaris
#bcast  le1 le2         # Solaris
#
```

#interfaz y nodo al que enviar los latidos

ucast eth0 h2

```
#mcast eth0 225.0.0.1 694 1 0
#
#       Set up a unicast / udp heartbeat medium
#       ucast [dev] [peer-ip-addr]
#
#       [dev]           device to send/rcv heartbeats on
#       [peer-ip-addr]  IP address of peer to send packets to
#
ucast eth0 192.168.15.181
#
```

la Ip que colocaremos sera la Ip del nodo contrario.

Si el nodo1 tiene Ip 192.168.15.180 entonces la configuración cruzada sera 192.168.15.181

Si el nodo2 tiene Ip 192.168.15.181 entonces la configuración cruzada sera 192.168.15.180

si el nodo cae y se recupera, vuelve a su rol original.

```
#       See the FAQ for information on how to convert
#               from "legacy" to "on" without a flash cut.
#               (i.e., using a "rolling upgrade" process)
#
#       The default value for auto_failback is "legacy", which
#       will issue a warning at startup.  So, make sure you put
#       an auto_failback directive in your ha.cf file.
#       (note: auto_failback can be any boolean or "legacy")
#
auto_failback on
#
```

esta configuración sera igual en los dos nodos

#servidor principal y secundario.

```
#
#watchdog /dev/watchdog
#
#       Tell what machines are in the cluster
#       node    nodename ...    -- must match uname -n
node    pc01
node    pc02
#
```

esta configuración sera igual en los dos nodos

Haresource: este fichero contiene información de los recursos que deseamos que tenga alta disponibilidad. Al principio de este archivo estableceremos quien será el servidor que predominará. La IP cirtual y el servicio que se brindará.

#nano /etc/ha.d/haresource

```
pc1@pc01: ~
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
  GNU nano 2.2.6              Fichero: /etc/ha.d/haresources

pc01 IPaddr::192.168.15.182/24/eth0 apache2

#
#       This is a list of resources that move from machine to machine as
#       nodes go down and come up in the cluster.  Do not include
#       "administrative" or fixed IP addresses in this file.
```

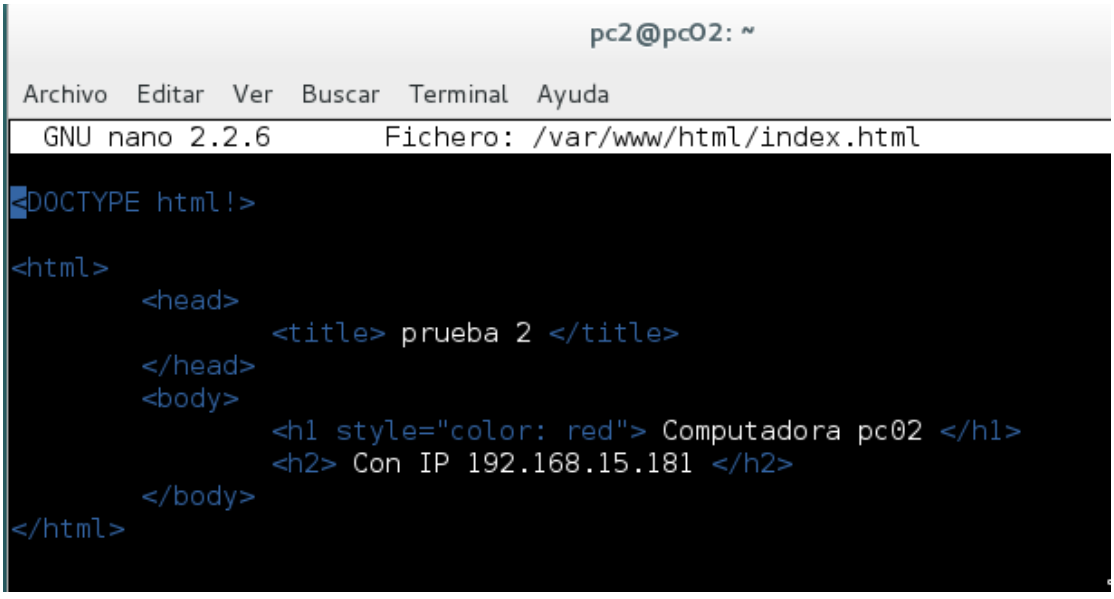
esta configuración sera igual en los dos nodos

luego modificamos el fichero index.html ubicado en /var/www/html/index.html

#nano /var/www/html/index.html

para tener una prueba gráfica del servicio.

Ilustración 8: Index, del servidor apache



Reiniciamos el servicio de HEARTBEAT en ambos nodos

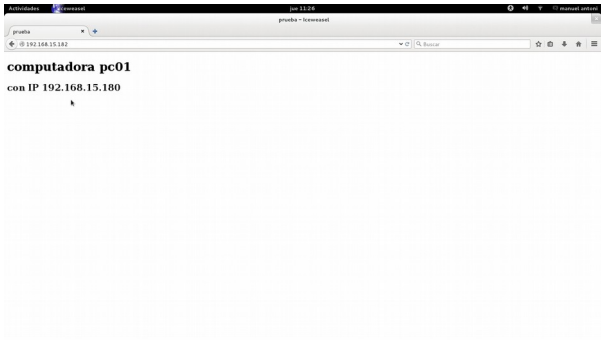
/etc/init.d/heartbeat stop

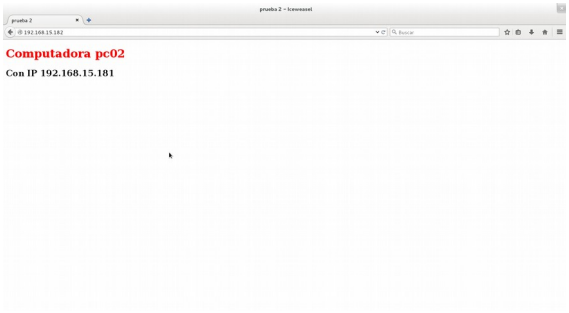
/etc/init.d/heartbeat start

luego encendemos las maquinas y levantamos el servicio

service heartbeat start

En ambas maquinas y listo.





Bibliografía.

Revista Digital Sociedad de la Información
http://www.sociedadelainformacion.com/13/cluster.pdf

Utilización y Administración avanzadas de sistemas GNU/Linux y aplicaciones Software Libre para estudiantes universitarios. Por José Angel de Bustos Pérez<jadebustos@augcyl.org>.

http://www.ibiblio.org/pub/linux/docs/LuCaS/Manuales-LuCAS/doc-curso-salamanca-clustering/html/clustering-ha.html#id2503903

Wikipedia. Título “Cluster Beowulf”
https://es.wikipedia.org/wiki/Cluster_Beowulf

Wikipedia. Título “Clúster (informática)”
https://es.wikipedia.org/wiki/Cl%C3%BAster_(inform%C3%A1tica)

Publicado en Informática e Internet Título “Alta disponibilidad que es y cómo se logra”
https://everac99.wordpress.com/2008/08/19/alta-disponibilidad-que-es-y-como-se-logra/

Cluster con Ultramonkey
http://www.estrellateyarde.org/virtualizacion/clusters-ultramonkey

Publicado por Sergio González “Alta disponibilidad”
http://www.sergio-gonzalez.com/doc/09-conceptos-de-clustering/html/cluster_alta_disponibilidad.html

ibiblio.org Biblioteca de Ciencias , de la Escuela de Periodismo y Comunicación de Masas , y Tecnología de la Información Servicios de la Universidad de Carolina del Norte en Chapel Hill Titulo “Clustering”
http://www.ibiblio.org/pub/linux/docs/LuCaS/Manuales-LuCAS/doc-curso-salamanca-clustering/html/ch03s04.html

Publicado En *debianhelp*, título “Loadbalanced High-Availability Apache Cluster Using Ultramonkey”
http://www.debianhelp.co.uk/ultramonkey.htm

Author: Falko Timme . TItulo “High availabity loadbanaced and apache cluster”
https://www.howtoforge.com/high_availability_loadbalanced_apache_cluster

Enciclopedia Colaborativa Canaima
http://wiki.canaima.softwarelibre.gob.ve/wiki/Balanceo_de_servidores_y_alta_disponibilidad_usando_Load_Directors