MECANISMOS DE ALTA DISPONIBILIDAD EN SERVIDORES

LUIS ALFONSO JEREZ JEREZ

DIANA CAROLINA RIVERA PEÑA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍA FÍSICOMECÁNICAS

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES

ESPECIALIZACIÓN EN TELECOMUNICACIONES

BUCARAMANGA
2011

MECANISMOS DE ALTA DISPONIBILIDAD EN SERVIDORES

LUIS ALFONSO JEREZ JEREZ

DIANA CAROLINA RIVERA PEÑA

Monografía presentada como requisito para optar al título de Especialista en Telecomunicaciones

Director

Mg. JORGE HERNANDO RAMÓN SUÁREZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍA FÍSICOMECÁNICAS

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES

ESPECIALIZACIÓN EN TELECOMUNICACIONES

BUCARAMANGA
2011

AGRADECIMIENTOS

Nuestros agradecimientos van dirigidos a:

La Santísima Trinidad y a la Santísima Virgen María por habernos permitido culminar esta etapa de nuestras vidas.

Nuestros familiares por todo su apoyo y ánimo brindado durante el transcurso de nuestros estudios.

Todos los docentes por su formación académica, paciencia, enseñanza y experiencia compartida.

Nuestro director Jorge Hernando Ramón Suárez por la colaboración y orientación dada durante el desarrollo de nuestra monografía.

Nuestros compañeros y auxiliares de clase por brindarnos su amistad y colaboración.

Todas las personas y/o instituciones que de alguna forma contribuyeron con sus aportes.

TABLA DE CONTENIDO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

ANEXOS

Pág.

RESUMEN

TITULO: MECANISMOS DE ALTA DISPONIBILIDAD EN SERVIDORES[*]

AUTORES: LUIS ALFONSO JEREZ JEREZ, DIANA CAROLINA RIVERA PEÑA[**]

PALABRAS CLAVES: Servidores, alta disponibilidad, servicios de red.

DESCRIPCIÓN

En la actualidad, los costos asociados a fallas en los sistemas de información pueden llegar a ser muy elevados, por lo que es necesario contar con arquitecturas de alta disponibilidad. Por lo tanto las arquitecturas de alta disponibilidad son una alternativa viable para la protección de datos y la eliminación de riesgos planificados y no planificados de caídas del sistema, evitando pérdidas económicas de producción y de datos.

Hoy en día la información de las empresas se maneja de forma sistematizada por lo que se pretende alcanzar un mayor rendimiento, oportunidad y continuidad en las labores empresariales. Con el fin de solventar las falencias en el manejo de la información, el presente trabajo de monografía muestra el estudio comparativo de los mecanismos de alta disponibilidad destacando sus principales características y elementos que lo integran. Así mismo, la configuración de servidores distribuidos en paralelo (clúster) para brindar mecanismos de alta disponibilidad en modo activo/pasivo ofreciendo los servicios: Web (Apache), DNS (Servicio de Resolución de Nombres), HTTPS (Protocolo de Transferencia de Hipertexto Seguro), FTP (Protocolo de Transferencia de Archivos) y base de datos. De manera que la implementación de un clúster de alta disponibilidad evidencia un incremento considerable en la prestación de los servicios, por lo que al presentarse una inoperatividad o falla en el servidor principal los demás servidores deben estar preparados para asumir inmediatamente los servicios, realizando en cualquier momento los diversos procesos que se están llevando a cabo, siempre de forma transparente para el usuario final.

Es importante resaltar que la mayoría de los aplicativos de alta disponibilidad funcionan bajo servidores Linux, los cuales se destacan por presentar muchos beneficios tecnológicos, económicos y académicos para pequeñas, medianas y grandes empresas.

[*] Trabajo de grado.

[**] Facultad Físico Mecánica. Escuela de Ingeniería Eléctrica, Electrónica y de Telecomunicaciones. Director: Ing. Jorge Hernando Ramón Suárez.

ABSTRACT

TITLE

MECHANISMS FOR HIGH AVAILABILITY IN SERVERS*

AUTHOR(S)

LUIS ALFONSO JEREZ JEREZ, DIANA CAROLINA RIVERA PEÑA**

KEY WORDS

Servers, high availability, network services.

DESCRIPTION

Currently, the cost associated with failures in information systems can be very high, making necessary to have high availability architectures.

Specifically, the servers where the data resides in the enterprise applications and services require continuous service availability. To achieve these objectives should consider the costs and risks which need implement those solutions. Therefore, the high availability architectures are a viable alternative for data protection and risk elimination of planned and unplanned downtime, avoiding loss of production and economic data.

Today the company information is handled in a systematic way so you will achieve a higher performance, opportunity and business continuity in the work. To address the shortcomings in information management this paper shows the work of comparative study of high availability mechanisms highlighting its key features and elements the comprise it. Also, the configuration of distributed servers in parallel (cluster) to provide high availability mechanisms in an active/passive service offering: Web (Apache), DNS (Name Resolution Service), HTTP (Hypertext Transfer Protocol Secure), FTP (File Transfer Protocol) and data base. So implementing a high availability cluster reveals a considerable increase in the provision of service, so by presenting a non-functioning or failure on the master server the servers should be ready for services immediately, making any the various processes currently being carried out provided transparently to the end user.

Importantly, most high availability applications running under Linux, which is notable for its many technological, economic and academic benefits for small, medium and large companies.

---

* Grade Project.

** Faculty of Physic Mechanical School of Electrical Engineering, Electronics and Telecommunications, directed Ing. Jorge Hernando Ramón Suárez.

1. INTRODUCCIÓN

Los sistemas a implementar en una empresa deben estar configurados y desarrollados con la mayor precisión para evitar que se presenten fallos en hardware y software generando grandes problemas.

Existen mecanismos que permiten evitar esta problemática como la implementación de equipos servidores en línea, interconectados, facilitando que fluya la información entre ellos, de tal manera que al presentarse una falla pueda darse una solución inmediata. Para ello podemos configurar una maquina con sistemas operativos distribuidos en paralelo y protocolos que faciliten la conexión, la transmisión y la navegación de la información brindando un soporte continuo y transparente.

La tecnología de clúster (Administración de Servidores), surge como solución para el manejo de la información generando una disponibilidad permanente.

Estos clúster son un conjunto de dos o más servidores interconectadas entre sí y que se comunican por un mismo canal [1]. De manera que, el concepto de clúster de alta disponibilidad se debe entonces al uso de configuraciones avanzadas de hardware y software en los componentes del sistema, proporcionando aplicaciones y recursos a los usuarios constantemente.

[1] ARQUÉS, S. J M., Huguet, C.M., & Galindo. M. E. (2008). Administración de servidores. En Administración de sistemas operativos en red (pp. 23 - 81) Barcelona: UOC

2. PLANTEAMIENTO DEL PROBLEMA

2.1. DESCRIPCIÓN DEL PROBLEMA

Es normal que un sistema informático falle por un problema de hardware o software, el inconveniente, es el grado de necesidad que requiere las instituciones o empresas en la utilización de dicho servicio. Cabe citar como ejemplos: entidades bancarias, hospitales, aeropuertos, universidades, bibliotecas y agencias de seguridad; de los cuales la no operatividad de la red puede ocasionar graves pérdidas económicas, de información y datos; generando a la vez el descontento de los usuarios y el desprestigio de la imagen corporativa.

Ante la creciente dependencia de los sistemas de información, la globalización de los mercados, el comercio electrónico, la alta competencia entre las compañías y los altos costes asociados a los tiempos de parada requiere que la probabilidad de error en los sistemas informáticos sea mínima. Lo cual se puede conseguir a través de la utilización de sistemas tolerantes a fallos o mediante sistemas de clúster de alta disponibilidad[2].

No obstante el diseño e implementación de un sistema tolerante a fallos suele ser exageradamente costoso, puesto que debe asegurar la redundancia de los componentes de su hardware y la permanente asistencia técnica, caracterizándose por ser una solución totalmente

[2] ARQUÉS, S. J M., Huguet, C.M., & Galindo. M. E. (2008). Administración de servidores. En Administración de sistemas operativos en red (pp. 23 - 81) Barcelona: UOC

dependiente de la empresa contratada. Mientras que los sistemas de clusters de alta disponibilidad están basados en la replicación de servidores, los cuales pueden ser incluso ordenadores normales. Además, son escalables, es decir, permite conectar más equipos a la red del clúster, configurarlos y mejorar el rendimiento del servicio.

Las evidentes ventajas que presentan los clusters de alta disponibilidad sobre los sistemas tolerantes a fallos lo hacen atractivo e interesante, pero ¿Qué mecanismos de alta disponibilidad existen para configurar servidores? y ¿Cómo implementar un clúster de alta disponibilidad, económico y útil para una solución empresarial?

2.2. OBJETIVOS

2.2.1. Objetivo General

Realizar un estudio comparativo de los mecanismos de alta disponibilidad en servidores con el propósito de proveer un criterio de selección en la utilización de los mismos.

2.2.2. Objetivos Específicos

- Describir las características de los mecanismos de alta disponibilidad.

- Instalar y configurar un sistema de alta disponibilidad en modo activo/pasivo para así realizar las respectivas pruebas ante posibles interrupciones del servicio.

- Comparar la respuesta a fallos de servidores de alta disponibilidad con respecto a servidores dedicados que carecen de dicha configuración.

2.3. JUSTIFICACIÓN

Hasta hace un tiempo se creía que el tema de alta disponibilidad estaba dirigido sólo a grandes empresas, sin embargo hoy día, existen muchas herramientas de software libre que facilitan la instalación y configuración de servidores de alta disponibilidad orientada a pequeñas y medianas empresas.

La implementación de un clúster de alta disponibilidad tiene como propósito restablecer en pocos segundos el servicio y mantener la integridad de los datos, de lo contrario, el sistema podría estar fuera de servicio durante horas. De aquí, la necesidad de conocer los mecanismos de alta disponibilidad para implementar servidores basados en Linux.

3. MARCO TEÓRICO

3.1. SERVIDOR

Es un sistema que ofrece recursos propios (datos, ficheros, aplicaciones, impresoras, discos) a disposición de otros ordenadores llamados clientes[3].

3.1.1. Tipos de Servidores

Los tipos de servidores más comunes son:

- **Servidor Web.** Es un ordenador encargado de ejecutar el programa servidor HTTP (Hyper Text Transfer Protocol), tiene la función de publicar sitios web en Internet y responder las solicitudes de los navegadores[4].

- **Servidor de base de datos MySQL.** Es un sistema de administración de base de datos relacional, ideal para crear base de datos con acceso a páginas web, realizar consultas en línea o almacenar datos.

[3] ARQUÉS, S. J M., Huguet, C.M., & Galindo. M. E. (2008). Administración de servidores. En Administración de sistemas operativos en red (pp. 23 - 81) Barcelona: UOC.

[4] BROCHARD J. (2006), Introduction. En J. T. Parra (Ed.), Internet information services (pp.8 - 13). ENI

- **Servidor DNS.** Es utilizado para asignar nombres a todas las máquinas de una red y a la vez traducir el nombre de dominio en una dirección IP, permitiendo que los equipos puedan ser localizados a través de la red de Internet[5].

- **Servidor FTP.** También conocido como servidor de transferencia de archivos permite mover archivos entre ordenadores.

3.2. CLÚSTER

Un clúster de ordenadores es un sistema distribuido en paralelo que consiste en dos o más servidores interconectados compartiendo recursos y se comporta como si se tratase de uno solo. Esta medida incrementa enormemente la disponibilidad de un sistema, no sólo ante fallos, sino también contemplando las necesarias actualizaciones periódicas o de mantenimiento que requiere el mismo[6].

---

[5] MARTINEZ J. et al. (2009). Servicio. En IPv6 para Todos: Guía de uso y aplicación para diversos entornos (pp.77-78) Argentina: Internet Society.

[6] ARQUÉS, S. J M. Huguet, C.M., & Galindo. M. E. (2008). Administración de servidores. En Administración de sistemas operativos en red (pp. 34 - 35) Barcelona: UOC

3.2.1. Tipos de clúster

Los clusters se clasifican en varios tipos[7]:

- Clúster de alto rendimiento (HPC- Hight Performance Cluster). Es un conjunto de servidores utilizado para ejecutar tareas que requieran alta capacidad computacional.

- Clúster de alta disponibilidad (HA- Hight Availability cluster). Está diseñado para proporcionar disponibilidad y mantener el buen funcionamiento del sistema mediante la utilización de hardware y software avanzado. Un clúster de alta disponibilidad presenta las siguientes características:

  - Detecta un fallo de hardware o software que se presente en los nodos.
  - Cuando el nodo principal falle, reinicia el servicio en cualquiera de los nodos restantes. En caso de volver a recuperarse el nodo principal, los servicios migran a este.
  - No necesita de un operador para mantener el servicio activo.
  - Garantiza la integridad de los datos.

[7] ARQUÉS, S. J M., Huguet, C.M., & Galindo. M. E. (2008). Administración de servidores. En Administración de sistemas operativos en red (pp. 23 - 81) Barcelona: UOC

- Clúster de alta eficiencia (HT- Hight Throughput). Está diseñado para ejecutar la mayor cantidad de tareas en el menor tiempo posible.

3.2.2. Modos de configuración de un clúster

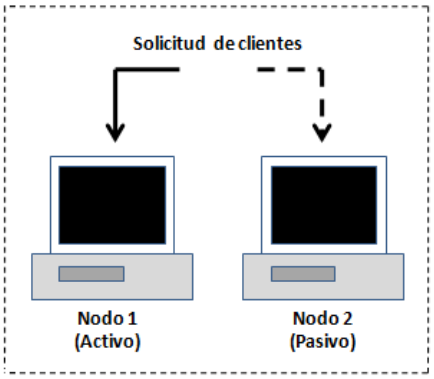
Los modos de configuración de un clúster son:

- Activo/Activo
- Activo/Pasivo

3.2.2.1. Clúster Activo/Activo

Se caracteriza por tener dos o más equipos configurados en estado activos y proveer cada uno de ellos por lo menos un servicio.

Figura 1. Clúster Activo/Pasivo

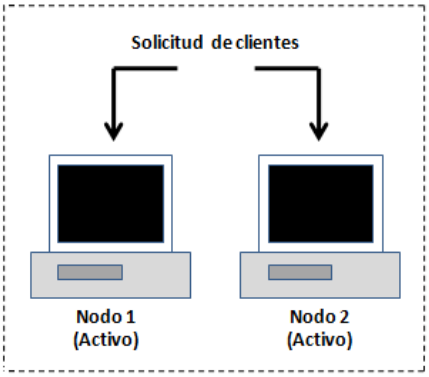


Fuente. Autores monografía

3.2.2.2. Clúster Activo/ Pasivo

Son dos equipos configurados para que cuando el nodo principal falle, el otro pueda suplirlo en la prestación del servicio.

Figura 2. Clúster Activo/Activo



Fuente. Autores monografía

3.3. ALTA DISPONIBILIDAD

La alta disponibilidad hace referencia a sistemas que operan sin interrupciones las 24 horas del día y durante los 7 días de la semana[8].

Normalmente un sistema de alta disponibilidad actúa sobre un sistema principal y otro de respaldo, permitiendo suplir cualquier incidencia ocasionada por razones humanas u operativas.

[8] Weygant P. S. (2001). Basic High Availability. EN D. Cullen-Dolce (Ed.),Cluster for high Availability, Prentice Hall (pp. 1 - 38) E.E.U.U.: Prentice-Hall, Inc.

3.3.1. Índice de alta disponibilidad

El índice de disponibilidad indica el porcentaje de tiempo que el sistema estuvo disponible durante un año, sin considerar la inoperatividad planificada por mantenimiento o actualización del sistema.

Tabla 1. Índice de disponibilidad

Índice de disponibilidad	Duración del tiempo de inactividad
97%	11 días
98%	7 días
99%	3 días y 15 horas
99,9%	8 horas y 48 minutos
99,99%	53 minutos
99,999%	5 minutos
99,9999%	32 segundos

Fuente. kioskea.net

La tabla 1 identifica el índice de disponibilidad, con su respectivo tiempo de inoperatividad, considerando que el sistema tendría 2 días de mantenimiento al año.[9]

[9] Índice alta disponibilidad (2011) Disponible en Internet: http://es.kioskea.net/contents/surete-fonctionnement/haute-disponibilite.php3

3.3.2. Medición de alta disponibilidad.

En un sistema real, si falla uno de los componentes, es reparado o sustituido por un nuevo componente. Si este nuevo componente falla, es sustituido por otro, y así sucesivamente[10].

Durante su vida útil, uno de los componentes puede ser considerado en Funcionando o en Reparación. El estado funcionando indica que el componente está activo y en reparación significa que ha fallado.

En forma simplificada, se dice que la disponibilidad de un sistema es la relación entre la duración de la vida útil de este sistema y de su tiempo total de vida. Esto puede ser representado por la fórmula:

Disponibilidad = MTTF / (MTTF + MTTR)

En la siguiente tabla se muestra el significado de las variables utilizadas para la medición de alta disponibilidad.

[10]Weygant Peter S. (2001). Cluster for high Availability, Prentice Hall ,Pag 17

Tabla 2. Variables de medición para el cálculo de HA

Nombre	Acrónimo	Cálculo	Definición
Tiempo Medio Entre Errores	MTBF	Horas/Número de errores	Duración media de funcionamiento de la aplicación antes de que produzca errores.
Tiempo Medio de Recuperación	MTTF	Horas de reparación/Número de errores	Tiempo medio necesario para reparar y restaurar el servicio después de que se produzca un error.

Fuente. http://msdn.microsoft.com

4. MECANISMOS DE ALTA DISPONIBILIDAD

La implementación de los mecanismos de alta disponibilidad se viene haciendo de dos maneras: mediante técnicas de rutado virtual (protocolo VRRP) o mediante mecanismos software implementados por encima del sistema operativo que detectan los fallos y toman las medidas necesarias.[11]

4.1. **VIRTUAL ROUTER REDUNDANCY PROTOCOL (VRRP)**

Es un mecanismo de alta disponibilidad donde se caracteriza por tener redundancia física en algunos componentes como router, switch, firewall y servidores.

Este Protocolo de redundancia de router virtual está diseñado para eliminar de forma automática el punto único de falla asociado a las redes de ruta estática, proporcionando conmutación por error mediante varias rutas de acceso LAN a través de routers alternativos.[12]

Características

- Establecido por el RFC 3768
- Utiliza la IP virtual y define automáticamente una MAC virtual para el clúster.

[11] Mecanismo de alta disponibilidad(2011) Disponible en Internet: http://www.ibiblio.org/pub/linux/docs/LuCaS/Presentaciones/200103hispalinux/ferrer/html/autenticidad-e-integridad.html

[12] VRRP(2011)) Disponible en Internet: http://www.cisco.com/en/US/products/hw/vpndevc/ps2284/products_tech_note09186a0080094490.shtml#intro

- Dentro del clúster se elige un Router activo y todos los demás permanecen como Routers de respaldo.

4.1.1. Estados de Los nodos (router virtuales)

- Master
- Backup

4.1.1.1. VRRP Master

El router adquiere el estado activo cuando responde a todos los requerimientos de la dirección IP. Puede haber solo un nodo MASTER en el router virtual, este envía paquetes de aviso a unos routers denominados BACKUP (usando dirección multicast) cada cierto tiempo configurándolo en la opción interval.

4.1.1.2. VRRP Backup

Es un estado cuando el MASTER se vuelve no-disponible o al menos 3 paquetes secuenciales VRRP se pierden, el cual genera un proceso de elección activando un nuevo MASTER basado en su prioridad.

4.2. MECANISMOS SOFTWARE

Son implementados por encima del sistema operativo, tales como el piranha, kimberlite, Linux- HA, los cuales permiten proveer la alta disponibilidad sin necesidad de invertir grandes cantidades en dispositivos hardware.

La alta disponibilidad se aplica a sistemas de información en las empresas, como una base de datos, equipos de comunicaciones (cortafuegos, servidores web, servidores de aplicaciones, etc.), los mecanismos aplicados que se empleen pueden ser distintos en función del entorno donde se vayan a implantar. Encontrándose así distintas configuraciones, como lo son:

- Activo - Pasivo. Se trata de disponer de un nodo funcionando, contando con todos los servicios que componen el sistema de información al que denominaremos Activo, y el otro nodo que se denominará Pasivo en el que se encuentran duplicados todos estos servicios, pero detenidos a espera de que se produzca un fallo.
- Activo - Activo. La configuración de alta disponibilidad en activo-activo es muy similar a la de activo-pasivo, aunque en este caso los dos nodos comparten los servicios de una manera activa, normalmente balanceados, consiguiendo una disponibilidad mayor ya que los servicios se entregan antes.

- Granja de servidores. Normalmente orientado a servicios web, servicios computacionales que se entregan de forma masiva, como puedan ser servicios terminales. En estas configuraciones no solo es importante la fiabilidad, también se hace indispensable contar con un sistema muy disponible por lo que se suelen colocar un gran número de máquinas haciendo una tarea común. Esta configuración siempre nos va a permitir que en caso de un nodo deje de hacer su función otro asuma su rol.

Tabla 3. Mecanismos de alta disponibilidad.

Mecanismos de Alta disponibilidad	
VRRP	**Aplicaciones software**
En términos de costo la configuración para aplicar a este mecanismo se debe instalar por lo menos en dos equipos, representando más inversión con los dispositivos hardware	No requiere licencias
Al mantener un ambiente GUI, lo cual proporcionar el entorno visual sencillo, donde permite la administración y la gestión en menor tiempo	Presenta Interfaces graficas menos amigables, Basados en comandos lo cual demanda más tiempo en operación
El soporte es continuo, por el costo de los dispositivos.	El soporte técnico es una desventaja debido a que es software libre, lo que puede representar al momento de una falla, el que no exista un respaldo o asesoría técnica rápida por parte del desarrollador de la aplicación.

Fuente. Autores monografía

El presente proyecto presenta la configuración de activo – pasivo para un clúster de alta disponibilidad (HA); utilizando el mecanismo de software implementado sobre el sistema operativo Fedora 15,

destacando las principales características de los componentes que integran al clúster HA.

4.3. COMPONENTES DE ALTA DISPONIBILIDAD EN UN CLUSTER HA

Los componentes necesarios para la instalación y configuración de un sistema de alta disponibilidad son:

4.3.1. Hardware

Son los componentes físicos que se utilizan para configurar la alta disponibilidad como:

- Servidores
- Memoria RAM
- Unidades de almacenamiento
- Unidad de backup

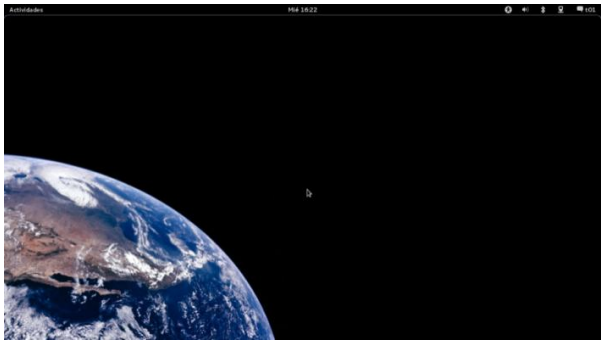
Las unidades de almacenamiento se puede clasificar en internos (discos instalados en las habías, que pueden ser 4 o más) o externos (arreglos de discos). Las capacidades disponibles para las unidades de backup pueden ir desde 12Gb hasta 800Gb o más.

4.3.2. Sistema operativo

Son un conjunto de programas encargados de administrar los recursos del computador a nivel de hardware y software, garantizando la ejecución de las diversas aplicaciones. Para la configuración se utilizó el sistema operativo Fedora 15, el cual se escogió por las siguientes características:

- Software libre y de código abierto, esto significa que puede ser utilizado, modificado y distribuido.
- Es fácil de usar, veloz y estable.
- Utiliza GNOME 3; siendo este aplicativo un programa diseñado para proporcionar un escritorio moderno[13], (Figura 3.), su primera versión fue publicada en 1999.

Figura 3. Entorno escritorio de GNOME 3



Fuente. Linux GNOME

[13] GNOME 3 made of easy (2011) Disponible en Internet: http://gnome3.org/index.html.es

Entre las características más destacables de GNOME 3 se tiene las siguientes:

- Intuitivo y elegante.
- Funciona sobre sistemas Linux.
- Facilita el acceso a ventanas y aplicaciones del equipo, e incluso mediante la utilización del teclado.
- Brinda mensajería integrada, esto significa que permite continuar conversaciones sin tener que cambiar de ventana.
- Crea áreas de trabajo.

4.3.3. Sistemas de Monitorización

La monitorización de los servicios juega un papel muy importante dentro del contexto de clúster de alta disponibilidad, puesto que por este medio se detecta continuamente el estado del sistema. Así por ejemplo, en el caso de presentarse alguna falla en el funcionamiento de alguno de los servidores, otro servidor tomaría lugar de este, recuperando rápidamente la prestación del servicio casi de manera transparente para el usuario.

Los mecanismos más utilizados para la monitorización de un sistema son: HeartBeat y Piranha.

4.3.3.1. HeartBeat[14]

Es un mecanismo de alta disponibilidad que se encarga de enviar paquetes de comprobación periódicamente, para verificar el correcto funcionamiento de los nodos mediante señales que se conocen como "latidos".

La diferencia entre el servidor principal y el servidor de respaldo radica en la prioridad que tienen los servidores para ofrecer el servicio. Así, el servidor de respaldo pasará a ofrecer el servicio sólo cuando deje de escuchar los latidos del servidor principal por un periodo predeterminado de tiempo, ya que entonces supondrá que éste ha dejado de funcionar. Una vez que el servidor de respaldo escuche los latidos del servidor principal, este tomara nuevamente el control.

La comunicación entre las dos máquinas se puede realizar por medio de las interfaces del puerto serial o la tarjeta Ethernet.

Los servicios más comunes que se pueden instalar en alta disponibilidad utilizando HeartBeat son: HTTP, DNS, NFS, Samba, LDAP, SMTP Y BBDD.

[14] Heartbeat (2011) Disponible en Internet: http://www.linux-ha.org/doc/users-guide/_heartbeat_as_a_cluster_messaging_layer.html.

Arquitectura de HeartBeat

En la arquitectura de HeartBeat se puede apreciar los siguientes elementos:

- **Módulo Gestor de los recursos del clúster o módulo CMR.** Es el encargado de arrancar los servicios, detenerlos, configurarlos, monitorizarlos y de establecer la configuración global del clúster. [15]

- **HeartBeat.** Se encarga del intercambio de mensajes para determinar la presencia o ausencia de pares de procesos entre los nodos.

- **Agentes de recurso.** Son un conjunto de scripts para la gestión y acceso a los diferentes recursos y servicios ofrecidos por el clúster.

- **ClusterGlue.** Permite la interoperabilidad entre sus diferentes elementos.[16]

4.3.3.2. Pirahna

Producto de Red Hat, el cual incluye un conjunto de herramientas útiles para monitorear un clúster de alta disponibilidad y realizar balanceo de carga, basado en un entorno web. [17]

[15] Arquitectura Heartbeat (2011) Disponible en Internet: http://clusterlabs.org/
[16] ClusterGlue (2011) Disponible en Internet: http://www.linuxha.org/wiki/Cluster_Glue
[17] Piranha 2011) Disponible en Internet: http://sources.redhat.com/piranha

Piranha se destaca por las siguientes características:

- Es una colección de programas diseñado para usuarios que deseen configurar los servicios de clúster en el entorno Linux.
- Se centra en el archivo de configuración: /etc/lvs.cf.
- Ofrece un daemon llamado LVS, encargado del control y la comunicación entre los componentes del clúster.
- Posee otro daemon llamado Nanny, encargado de realizar el proceso de monitorización entre ambos nodos y reemplazar el equipo primario en caso fallas en el servicio y el daemon pulse, encargado de devolver el servicio primario si este se ha restablecido.

Los componentes principales de Pirahna son:

- **Nanny.** Encargado de realizar el proceso de monitorización entre ambos nodos y reemplazar el equipo primario en caso fallas en el servicio.
- **Pulse.** Encargado de devolver el servicio primario si este se ha restablecido.

4.3.4. Sistemas de Sincronización

El proceso de sincronización de la información consiste en la replicación de ficheros en todos los ordenadores, con el fin de garantizar la integridad de los datos.

Entre los sistemas de sincronización más utilizados se encuentran: Rsync y Unison.

- **Rsync.** Es un programa que permite copiar archivos entre dos sistemas UNIX, mediante la utilización de un algoritmo, sincronizando los archivos que fueron modificados. Rsync permite:

  - Copiar ficheros y directorios.
  - Redireccionar todo el tráfico a través de SSH para cifrarlo.
  - Permitir acceso a terceras personas para que puedan hacer un espejo de las páginas.
  - Es unidireccional, permite mantener una réplica de un directorio.
  - Actualizan novedades de los ficheros y no permite modificar la réplica.

- **Unison.** Este programa permite sincronizar archivos y carpetas tanto localmente como de manera remota entre diferentes equipos.

  - Tiene disponibles varias versiones para sistemas operativos como Linux, Windows y otros.
  - Permite guardar en diferentes máquinas o diferentes discos.
  - No necesita privilegios de administrador, acceso al sistema o cambios en el núcleo para funcionar.
  - Trabaja a través de internet o en la propia máquina.
  - Permite usar protocolos seguros como SSH.
  - La sincronización se lleva a cabo de manera bidireccional y puede controlarse a través de un intuitivo frontal gráfico.

4.3.5 Bases de datos

Es un sistema de administración y gestión de datos, donde se encuentra almacenada la información del sistema web, para la configuración de alta disponibilidad se eligió el gestor de base de datos MySQL.

- **MySQL 3. 41.** Es un sistema de administración y gestión de base de datos relacional que permite administrar los datos en un entorno web, especialmente con arquitecturas cliente servidor, además es compatible con varios lenguajes de programación[18]. Las principales características de este gestor de datos son las siguientes[19]:
  - Fácil instalación y configuración.
  - De distribución gratuita y puede ser modificado.
  - Permite descargar su código fuente, lo cual favorece su desarrollo y continuas actualizaciones.
  - Alto rendimiento, fiabilidad y facilidad de uso.
  - Dispone de APIs (Applications Programming Interface) en una gran cantidad de lenguajes (C, C++, Java, PHP, entre otros).
  - Permite aplicaciones basadas en la pila LAMP (Linux, Apache, MySQL, PHP/Perl/Python).
  - Gran portabilidad entre sistemas.

[18] THIBAUD C. (2006). Presentación de MySQL. En MySQL 5 (pp.6) Barcelona :ENI

[19] Mysql(2011) Disponible en Internet: http://www.mysql.com/

- Se ejecuta en más de 20 plataformas incluyendo Linux, Windows, Mac, Solaris, HP-UX, IBM AIX.
- Soporta grandes cantidades de tipo de datos para las columnas.
- Soporta hasta 32 índices por tabla.
- Gestión de usuarios y contraseñas, generando un buen nivel de seguridad.

Sin embargo una de las desventajas de MySQL es que carece de una interfaz gráfica para realizar su trabajo, por lo que todas sus consultas las debe realizar por líneas de comando.

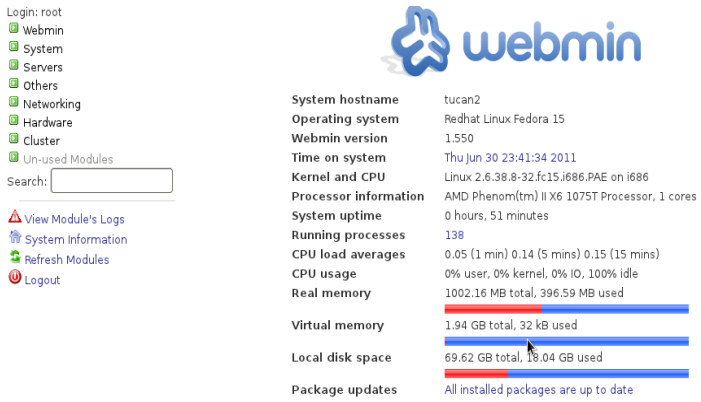
4.3.6 Aplicativos

Son los programas utilizados dentro de la configuración del clúster de alta disponibilidad, para administrar o gestionar algunos servicios.

4.3.6.1. Webmin 1.550

Es una herramienta remota con una interfaz web para la administración y configuración de servicios, como el servidor Web (Apache), MySQL, DNS, entre otros. Se ingresa por cualquier navegador, el cual permite establecer una conexión http mediante el puerto 10000 por defecto.

Figura 4. Pantalla Webmin



Fuente. Aplicativo Webmin

Después de su instalación (ver anexo 2), esta aplicación permite administrar y monitorear el sistema se puede crear usuarios y grupos, se puede observar que procesos se están ejecutando, programar la ejecución de trabajos, reiniciar o apagar el equipo. También se tiene control de los servidores Apache, DNS, MySQL entre otros. Además permite configurar y monitorear un clúster.

4.3.6.2. PhpMyAdmin

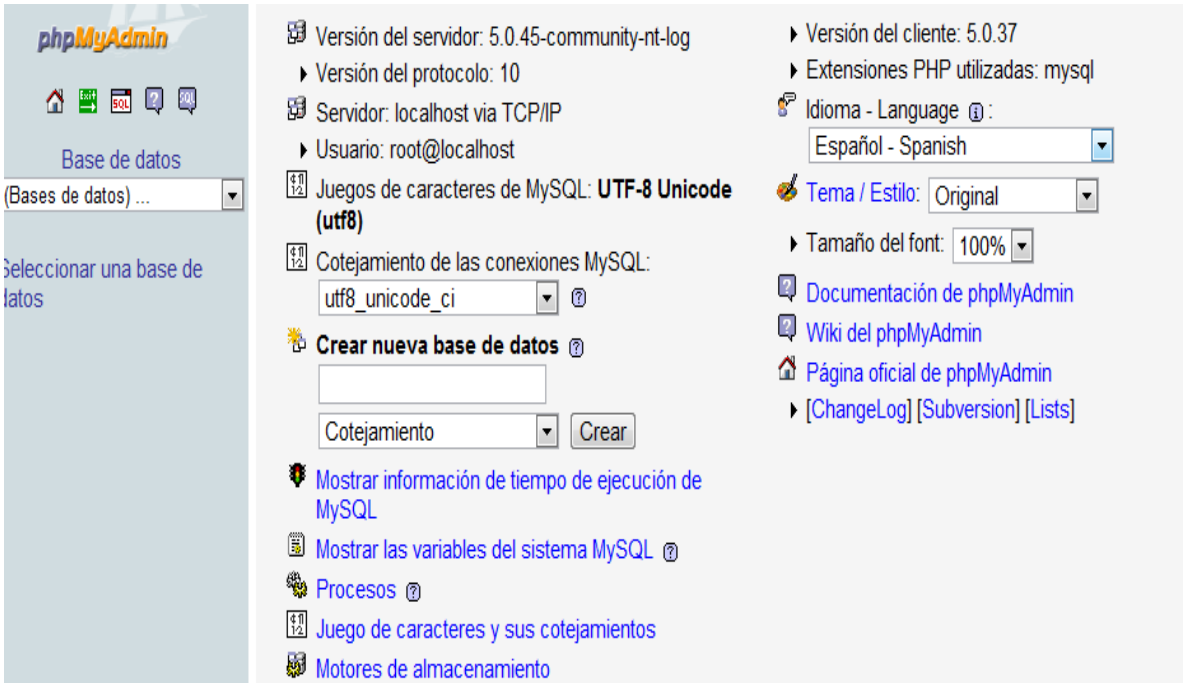
Es una herramienta de software libre desarrollada en PHP, para gestionar la administración de MySQL,[20] son cuantiosos las ventajas de esta aplicación, algunas de ellas son la gestión de la base de datos del servidor MySQL, facilita los diferentes objetos de una base de datos, manejando tablas, columnas, índices, vistas, entre otros aspectos.

[20] HEURTEL, O.(2009) Introducción a MySQL. En D. Marin (Ed.), PHP y MySQL Domine el desarrollo de un sitio Web Dinámico e interactivo (pp. 15 - 43) Barcelona : ENI

Algunas de las características que destacan a PhpMyAdmin, son:

- Permite crear, eliminar y alterar base de datos.
- Elimina, edita y agrega campos.
- Administra campos claves.
- Ejecuta sentencias escritas en lenguaje SQL.
- Administra usuarios con diferentes privilegios.
- Exporta datos a diferentes formatos.
- Administra múltiples servidores.

Figura 5. PhpMyAdmin



Fuente. Aplicativo PhpMyAdmin

Por lo tanto, PhpMyAdmin permite trabajar una base de datos de manera visual y estructurada.

4.4. BALANCEO DE CARGA

Clúster que comparte la carga de trabajo y las peticiones que realizan los clientes entre varios nodos, esto permite mejorar el tiempo de respuesta, acceso y confiabilidad.

Un balanceador de carga realiza las siguientes funciones:

- Fracciona el tráfico en peticiones individuales.
- Decide cual servidor recibe la petición.
- Mantiene un monitoreo en los servidores y verifica que estén respondiendo.

El balanceo de carga puede ser mediante software o hardware.

4.4.1. Balanceadores software

- **IPVS (Servidor virtual IP)**. Implementa balanceo de carga a nivel de la capa de transporte dentro del kernel de Linux (layer4-switching)[21]. Puede redirigir conexiones a servicios TCP/UDP a los servidores reales, y hace que los servicios de los servidores reales se muestren como un servicio virtual de única dirección IP.

  Características:

  - Soporte para protocolos UDP y TCP
  - Tres métodos de reenvio de paquetes: NAT, Tunneling, Direct Routing.

[21] IPVS (2011) Disponible en Internet: web:http://www.linuxvirtualserver.org/software/ipvs.html

➢ Ocho algoritmos de balanceo de carga: Round robin, weighted round robin, least-connection, weighted least-connection, locality-based least-connection, locality-based least-connection with replication, destination-hashing, and source-hashing.

- **Keepalived** [22]**.** Permite verificar si uno de los nodos está caído, keepalived le informa al kernel linux por medio de una llamada setsockopt para que remueva la entrada de este servidor de la topología LVS.

- **Piranha**. Es estrictamente una aplicación de software de HA. Piranha es el nombre del paquete de cluster, y también de la herramienta GUI de administración para el cluster.[23]

4.4.2 Balanceadores hardware

Son dispositivos físicos con el propósito de balancear cargas entre los clúster. Las principales ventajas de los balanceadores, es la potencia, estabilidad y escalabilidad; las desventajas son el costo y el mantenimiento.
Algunas de estas soluciones hardware de balanceo de carga son:

- **Barracuda Load Balancer.**[24] Está diseñado para proveer completas capacidades de balanceo de carga de IP para

[22] Keepalived (2011) Disponible en Internet: http://www.keepalived.org

[23] piranha (2011) Disponible en Internet: http://www.redhat.com/support/wpapers/piranha/x32.html

cualquier aplicación sobre la base de IP, incluso: Los sitios de Internet con altos requerimientos de tráfico, incluso Web, FTP, difusión multimedia y redes de entrega de contenidos.

Figura 6. Barracuda Load Balancer

Fuente. Página Web Barracuda Networks

- **El F5 BIG-IP Local Traffic Manager.**[25] Realiza balanceo de carga entre los servidores en un solo centro de datos. Utiliza la topología de equilibrio de carga basado para inspeccionar IP de un usuario y determinar el centro de datos más eficiente.

Figura 7. F5 BIG-IP Local Traffic Manager

Fuente. Página Web F5 Big I

[24] Barracuda Load Balancer (2011) Disponible en Internet:http://www.barracudanetworks.com/ns/products/balancer_overview.php?L=es

[25] El F5 BIG-IP Local Traffic Manager (2011) Disponible en Internet: http://www.f5.com/glossary/load-balancing.html

5. CONFIGURACION DE SERVICIOS DE RED Y DE APLICATIVOS DE ALTA DISPONIBILIDAD

Para la configuración de los servicios de red se instalaron previamente los siguientes componentes a nivel de software:

Sistema operativo: Fedora 15.
Servidor Web: Webmin 1.550 que proporciona: Apache, PHP, DNS y el cliente MySQL.
Base de datos: MySQL 3.41.
Alta disponibilidad: HeartBeat y Rsync.

5.1 INSTALACIÓN DE SERVICIOS

El proceso de instalación de los diferentes servicios se realizó desde la consola de comandos ingresando como usuario root.

5.1.1. Servicio httpd

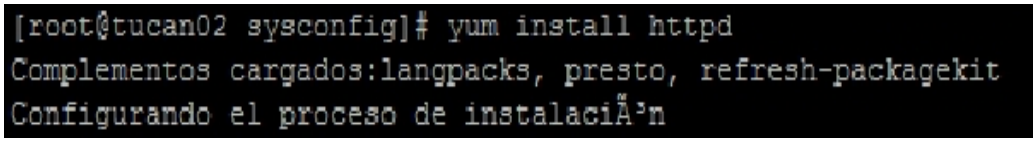
También conocido como servidor apache, es de distribución libre y de código abierto, se encarga de responder las peticiones de los navegadores mostrando las páginas web solicitadas[26].

5.1.1.1. Instalación httpd

Para instalar el servicio http se ejecuta desde la consola el comando:
yum install httpd

[26] Castillo f (2010). Servicio en red (pp.150-155) España: ed. nobel

Figura 8. Instalación httpd



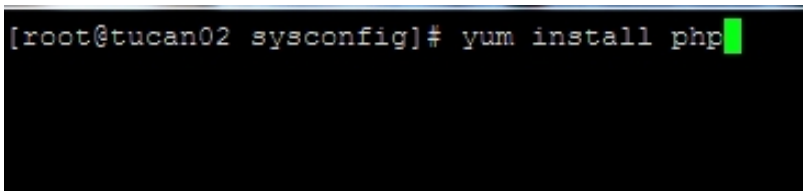
Fuente. Autores monografía

5.1.2. Instalación de PHP

El proceso de instalación de PHP se realiza de la siguiente manera:

yum install php

Figura 9. Instalación PHP

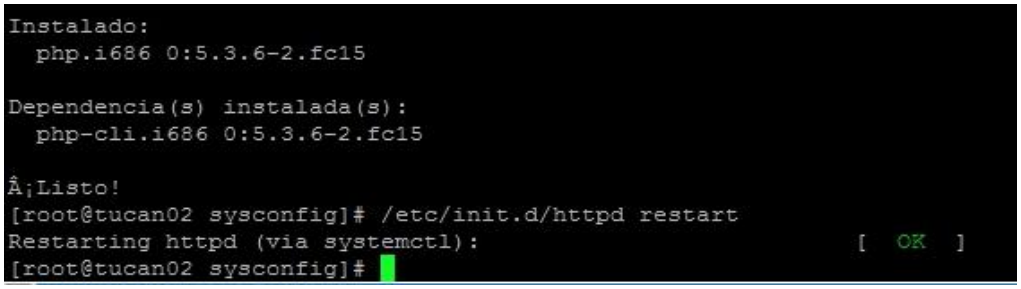


Fuente. Autores monografía

Para finalizar la descarga e instalación de PHP, se debe reiniciar el servicio mediante el comando:

/etc/init.d/httpd restart

Figura 10. Reiniciar el servicio httpd



Fuente. Autores monografía

5.1.3. Instalación MySQL

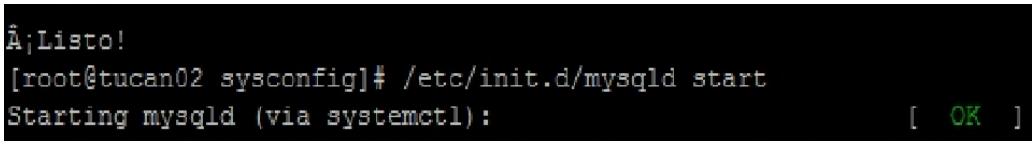
Desde la consola se ejecuta el comando:

yum install mysql mysql-server

Luego de instalado, se inicia el servicio MySQLd mediante el siguiente comando:

/etc/init.d/mysqld start

Figura 11. Iniciar servicio MySQLd



Fuente. Autores monografía

5.1.4 Instalación PhpMyAdmin 3.4

Desde la consola se ejecuta como usuario root el siguiente comando:

yum install phpmyadmin system–config-httpd

Una vez instalado el paquete PhpMyAdmin se debe reiniciar apache mediante el comando:

/etc/init.d/httpd restart

5.1.5 Servicio FTP

Se utiliza para transferencia de archivos desde un equipo local a un servidor remoto o local mediante un cliente FTP.

Un cliente FTP es un programa que permite conectarse con el servidor, este requiere un nombre de usuario y una contraseña para identificarse como usuario con permiso para recibir datos de este servidor[27], además existen otro tipo de usuarios en FTP como el usuario anónimo.

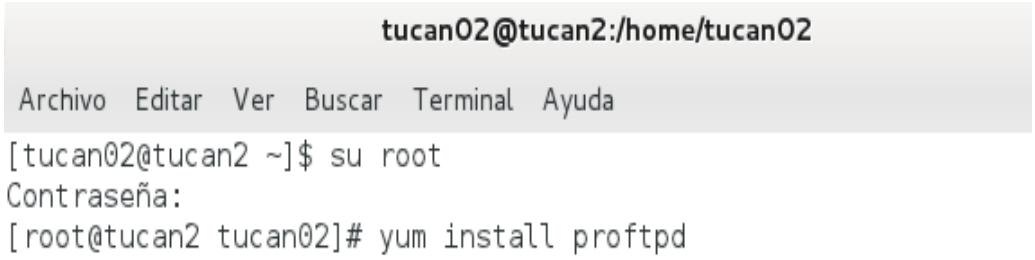
[27] LACKERBAUER I. (2009). La Descarga De Un Archivo En L. Fontana, (Trad.). Internet Todo Sobre Internet.(pp. 125 -13). España: BOIXAREU(Trabajo original publicado en 2000)

5.1.5.1 Instalación del servicio FTP

Existen diversos servidores FTP, en este caso se escogió la instalación del servicio Proftpd. Para la instalación se ejecutó el comando:

yum install proftpd

Figura 12. Instalación FTP



Fuente. Autores monografía

Se inicia el servicio mediante el comando:

/etc/init.d/proftpd start

5.1.6 BIND

BIND (Berkeley Internet Name Domain) se originó en la universidad de California y es la implementación del protocolo DNS, el cual provee componentes del sistema de nombre de dominio, este incluye un servidor de nombres de dominio (Named) [28].

[28] Bind (2011) Disponible en Internet: http://www.isc.org/software/bind/whatis

La distribución del software BIND contiene tres partes:

- Nombre de dominio del servidor del sistema. Es un programa llamado named que responde a las peticiones que se le envían.

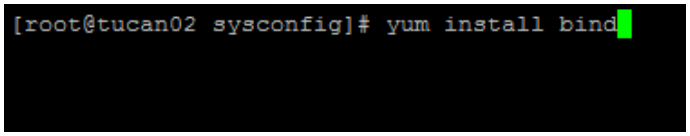
- Nombre de dominio de librería de solución. Es un programa que resuelve las preguntas a los servidores apropiados y responde adecuadamente las peticiones de los servidores.

- Herramientas de software para servidores de pruebas. Son herramientas se utilizan para determinar si el servidor está funcionando correctamente.

5.1.6.1 Instalación BIND

Para la instalación se ejecuta el comando:
yum install bind

Figura 13. Instalación Bind



Fuente. Autor monografía

Después de instalado, se inicia el servicio mediante el comando:

/etc/init.d/named start

5.2 CONFIGURACIÓN DE ALTA DISPONIBILIDAD

Para la configuración del sistema de alta disponibilidad se utilizaron a nivel de hadware dos equipos.

Equipo 1. Se le asignó el nombre de tuca01 y funcionará como nodo principal. Las características que presenta este equipo son:

Marca Hewlett-Packard
Procesador Intel I3
Memoria RAM de 3 Gb
Disco duro 320 Gb
Tarjeta de red Ethernet

Equipo 2. Se le asignó el nombre tucan02 y funcionará como nodo de respaldo. Las características que presenta este equipo son:

Marca Toshiba
Procesador Intel core I5
Memoria RAM de 4Gb
Disco duro 500 Gb
Tarjeta de red Gigabit Ethernet 10/100/1000mb 64bits

Los servicios instalados en los dos servidores son httpd, MySQL, DNS, y FTP y su funcionamiento en el esquema de alta disponibilidad es activo-pasivo salvo MySQL que funciona activo-activo en los dos nodos.

Se configuro una dirección ip a cada uno de los servidores en la interfaz de red Ethernet así:

Tucan01 192.168.1.121

Tucan02 192.168.1.129

5.2.1 Instalación de HeartBeat

La instalación del HeartBeat se ejecuta con el comando:

yum install heartbeat

5.2.2 Archivos de configuración del HeartBeat

La configuración de HeartBeat consta de 3 archivos, estos son: Authkeys, ha.cf y haresources.

- Authkeys[29]. Es un archivo de autenticación donde se almacena la clave que permite establecer la comunicación entre los nodos. Hay tres opciones de autenticación que son: CRC, md5 y sha1. (Para visualizar la configuración ver anexo 3).

[29] Heartbeat (2011) Disponible en Internet: http://www.linux-ha.org/doc/man-pages/re-authkeys.html

- Ha.cf[30]. En este archivo se definen los nodos y la interfaces de HeartBeat. Los parámetros modificados para la instalación de este sistema de alta disponibilidad son:

a. Debugfile/var/log/ha-debug (Ruta donde se almacena el debug de HeartBeat).

b. logfile /var/log/ha-log (Ruta donde se almacena el log de HeartBeat).

c. logfacility local0 (Establece el nivel del log).

d. keepalive 2 (Especifica el tiempo en que HeartBeat enviará paquetes, para comprobar la disponibilidad de los nodos. En este caso cada 2 segundos).

e. deadtime 30 (HeartBeat confirmará la caída del nodo cada 30 segundos).

f. warntime 10 (HeartBeat avisará la falla de un nodo cada 10 segundos).

g. initdead 120 (Tiempo máximo en que HeartBeat esperará a que un nodo arranque, 120 segundos).

h. udpport 694 (Puerto UPD para la comunicación unicast).

[30] Heartbeat (2011) Disponible en Internet: http://www.linux-ha.org/ doc/man-pages/re-hacf.html

i. bcast em2 (Interfaz para el broadcast).

j. ucast em2 192.168.1.129 (Dirección IP unicast e interfaz de red).

Este último parámetro es diferente en cada equipo. Para el nodo1 (tucan01), se le asigna la IP del segundo nodo y para tucan02 se le asigna la dirección IP del primer nodo ucast em2 192.168.1.121).

k. auto_failback on (Se indica al HeartBeat que cuando se restablezca el nodo principal, este retorne el control del servicio. Si este parámetro se deja en off el servicio no retorna al nodo principal hasta que el nodo secundario falle).

l. node tucan01 node tucan02 (Nombra los nodos activos).

Para observar la configuración ha.cf, ver anexo 4.

- Haresorces. En este archivo se le indica cual es el nodo principal, se configura la IP virtual flotante y se especifican que servicios se monitorean para ver la configuración de este archivo (ver anexo 5).

tucan01 IPaddr2::192.168.1.10/24/em2 httpd named FTP

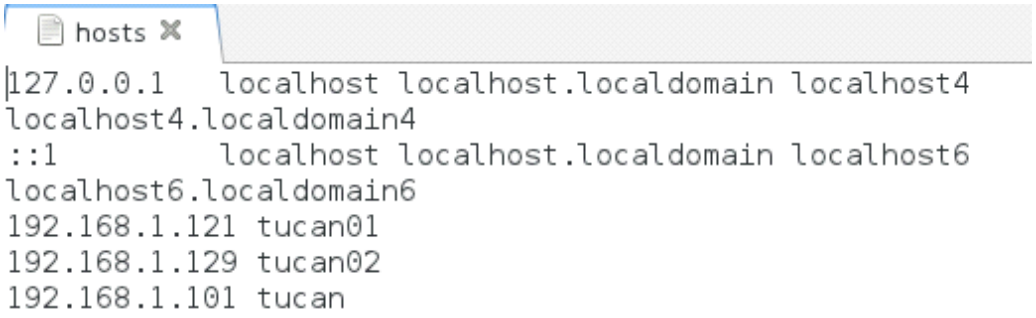
En la configuración del HeartBeat se registra la IP válida para habilitar la interfaz de red virtual flotante. Cuando el servidor tucan01 (nodo principal) se encuentre activo, esta dirección IP flotante estará asignada al equipo, al presentarse un fallo en el nodo principal, el HeartBeat lo

detecta e inicia el proceso para habilitar los servicios pasivos en el nodo secundario y la dirección IP flotante quedará apuntando a tucan02.

La dirección IP virtual asignada fue la 192.168.1.101, la cual se escogió de manera aleatoria para la interfaz de red virtual flotante.

Para finalizar la configuración de HeartBeat se modifica el archivo hosts de las máquinas, con el fin de indicar la dirección IP que representa cada equipo, como se observa en la figura 12.

Figura 14. Host



Fuente. Autores monografía

Una vez realizada la configuración en ambos servidores, se inicia el proceso HeartBeat.

Tucan01# /etc/init.d/heartbeat start
Tucan02 # /etc/init.d/heartbeat start

5.3 COMPROBACIÓN DE LA CONFIGURACIÓN

Una vez configurado el HeartBeat, se comprueba el funcionamiento del sistema de HA ingresando en el navegador la dirección IP flotante (192.168.1.101). El nodo principal debe responder con el servicio httpd activo.

5.4 SINCRONIZAR BASES DE DATOS

Después de haber instalado los servicios de httpd y mysqld, se inicia la sincronización de las bases de datos mediante la configuración de una réplica de MySQL-server en tucan01 a MySQL-server en tucan02. La replicación indica que los cambios que se realicen en tucan01 quedan reflejados en tucan02, en consecuencia, ante una falla del servicio en tucan01, los datos se encontrarán en tucan02.

Es muy importante tener en cuenta que cuando el servicio se restablece es necesaria la actualización manual de la base de datos en tucan01. La cual no se realizó en forma bidireccional porque para ello, sería necesario utilizar 3 servidores como mínimo, un administrador y dos nodos.

La configuración de la réplica se hizo así:

Para el archivo de configuración de myslq my.cnf en el nodo principal (tucan01) se incluyeron las siguientes líneas:

Log-bin=MySQL-bin (Configura el archivo binario con la información de los cambios).

Binlog-ignore-db=test (Excluye la base de datos test de la réplica).

Server-id=1 (Asigna un ID al servidor, mayor de 0).
Enseguida se reinicia el servicio mediante el comando:

/etc/init.d/MySQLd restart

Para el archivo de configuración MySQL my.cnf en el nodo secundario (tucan02) se incluyó la siguiente línea:

server-id=2 (ID del servidor)

Se reinicia el servicio.

A continuación se crea un usuario para la replicación en el nodo principal (tucan01) desde la consola de MySQL.

#MySQL –u root –p
Password:
MySQL>GRANT REPLICATION SLAVE ON *.* TO 'replica'@tucan02 IDENTIFIED BY 'slave'; (se crea desde el master el usuario replica para el esclavo (tucan02), se le asigna contraseña slave y se le conceden permisos de réplica a este usuario).

Query OK, 0 rows affected (0.04 sec)

MySQL> SHOW MASTER STATUS (se pide el estado del master para conocer el log del archivo binario y la posición, esta información se necesita para configurar el esclavo).

```
+-----------------+----------+--------------+------------------+
| File            | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+-----------------+----------+--------------+------------------+
| MySQL-bin.000001 |      107 |              | test             |
+-----------------+----------+--------------+------------------+
1 row in set (0.00 sec)
MySQL>quit
```

Desde la consola de MySQL en el esclavo (tucan02)

#MySQL –u root –p

Password:

MySQL>CHANGE MASTER TO MASTER_HOST='tucan01', MASTER_USER='replica', MASTER_PASSWORD='slave', MASTER_LOG_FILE='MySQL-bin.000001', MASTER_LOG_POS=107, MASTER_PORT=3306; (Se le indica al esclavo que el master es tucan01, cual es el usuario y su contraseña para realizar las réplicas, el MASTER_LOG_FILE Que se observó en la consulta en el master, EL MASTER_LOG_POS, posición en el archivo binario que también aparece en la consulta en el master y el puerto por el cual se comunican)

Query OK, 0 rows affected (0.03 sec)

MySQL>START SLAVE;

MySQL>quit

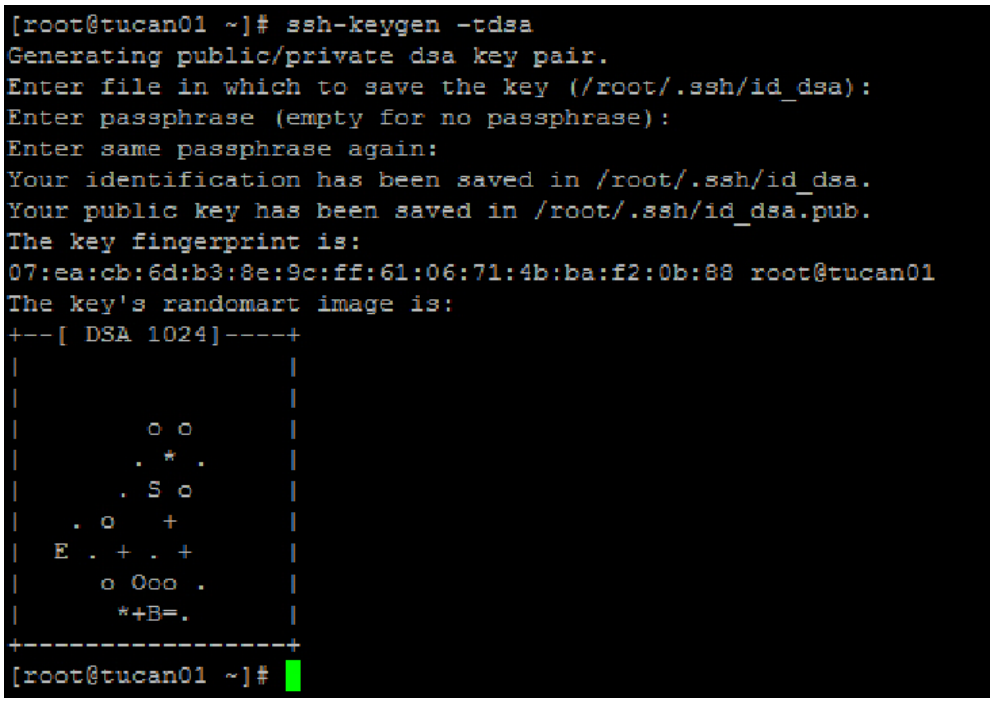
En este punto queda configurada la réplica de tucan01 en tucan02.

5.5 SINCRONIZACIÓN DE LOS ARCHIVOS

Para la sincronización de archivos de la carpeta home se usó rsync, el cual hace una comparación de los directorios y sincroniza los archivos nuevos, actualizándolos de acuerdo a la fecha de creación o modificación.

La sincronización se realiza por medio de SSH, y por ser un proceso que se debe realizar de forma automática, se creó y configuro una llave pública en ambos servidores para permitir una comunicación transparente entre los dos servidores.

Figura 15. Creación de llave



Fuente. Autores monografía

Luego de generada la llave publica se copia al otro servidor para que la autenticación entre los dos servidores por ssh sea transparente (sin pedir contraseña).

Nodo tucan 01

#cat id_dsa.pub | ssh root@tucan02”cat /root/.ssh/authorized_keys”

Nodo tucan 02

#cat id_dsa.pub | ssh root@tucan01”cat /root/.ssh/authorized_keys”

Para configurar la llave pública en ambos servidores se creó una tarea de Cron desde el Webmin. Esta tarea ejecuta la instrucción de rsync que hace la sincronización de archivos como se observa en la figura 16. La configuración se realizó en ambos servidores para mantener sincronizados los archivos de la carpeta home, este proceso se realiza cada minuto durante las 24 horas mientras los nodos se encuentren activos.

Figura 16. Comando rsync

Fuente. Autores monografía

El parámetro –e ssh indica que la transferencia se realiza por ssh, -a mantiene todos los permisos, grupos, usuarios y fecha de los archivos, v muestra un detalle de la operación, /home/ root@tucan02:/home/

es la carpeta de destino.

Para tucan01 se modificó solo el destino (root@tucan01:/home/).

5.6 SINCRONIZACIÓN HTML

El comando para la sincronización de la carpeta html se muestra en la figura 17.

Figura 17. Comando de sincronizar carpeta home

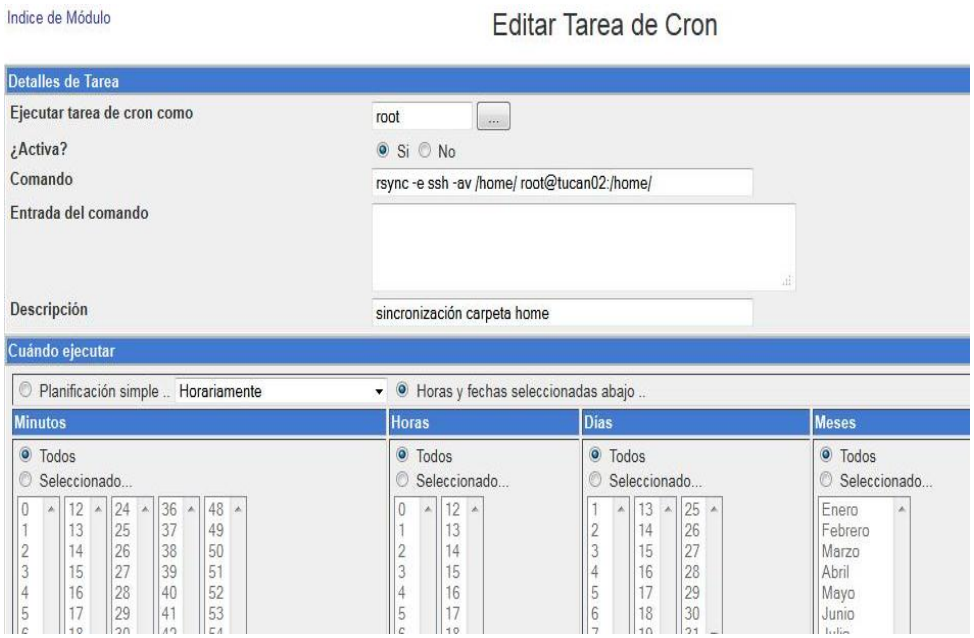
```
rsync -e ssh -av /var/www/html/ root@tucan01:/var/www/html/
```

Fuente. Autores monografía

La instrucción para la sincronización en el nodo 2 se muestra en la figura 18; para tucan02 el cron se ejecuta cada minuto durante las 24 horas.

Se podría sincronizar más carpetas, pero esto depende de la necesidad particular de cada sistema que se esté configurando.

Figura 18. Webmin Tarea de Cron



Fuente. Autores monografía

6. PRUEBAS DE ALTA DISPONIBILIDAD

En una red típica, una computadora de la red actuará o se definirá dentro de la topología de la misma como un servidor, el cual será un dispositivo de almacenamiento central para los archivos o programas compartidos sobre la red, cuando la red cuenta con un servidor dedicado, este equipo proporciona un punto central para las tareas de administración, tales como respaldo de archivos, administrar las comunicaciones, administrar los recursos de impresión, entre otros. Sin embargo, al presentarse un fallo en el servidor dedicado, la administración de la red, así como el funcionamiento de los servicios instalados en esta máquina se detendrían, donde esta recuperación podría demorar desde unos minutos, horas o días dependiendo de la complejidad del percance.

Entre los posibles inconvenientes que se pueden encontrar están los siguientes:

- Corte del fluido eléctrico.
- Daños de red (cableado y dispositivos).
- Calentamiento de dispositivos.
- Saturación del servidor por múltiples visitas.
- Ataque de negación del servicio.
- Fallas de algún componente de software.
- Errores de configuración.
- Daños intencionales.

Con el propósito de comparar la respuesta a fallos de un sistema de alta disponibilidad con respecto a un servidor dedicado se realizaron algunas pruebas.

6.1 DAÑO DEL PATCH CORD

Para el desarrollo de la prueba se utilizó un clúster HA en modo activo-pasivo compuesto por dos nodos (tucan01 y tucan02). El nodo principal (tucan01), ofrece los servicios httpd (activo-pasivo), MySQL (activo-activo), DNS (activo-pasivo) y FTP (activo-pasivo). Este nodo (tucan01) se encarga de atender las peticiones cuando el sistema se encuentra en funcionamiento normal. Al presentarse una falla en el nodo principal, el segundo nodo (tucan02) detecta la falla y asume el control de los servicios prestados.

Al presentarse un daño en el patch cord del nodo principal tucan01, a las 12: 46:09 hora local, se revisó el log del HeartBeat para comprobar el tiempo de respuesta y la restauración del servicio mediante la intervención del servidor de respaldo tucan02, registrándose la continuidad del servicio a las 12:48:59.

Figura 19. Log del sistema

```
Jul 07 12:46:09 tucan01 ipfail: [2596]: info: We are dead. :<
Jul 07 12:46:09 tucan01 ipfail: [2596]: info: Asking other side for ping node count.
Jul 07 12:48:59 tucan01 heartbeat: [2027]: CRIT: Cluster node tucan02 returning after partition.
Jul 07 12:48:59 tucan01 heartbeat: [2027]: info: For information on cluster partitions, See URL: http://linux-
Jul 07 12:48:59 tucan01 heartbeat: [2027]: WARN: Deadtime value may be too small.
Jul 07 12:48:59 tucan01 heartbeat: [2027]: info: See FAQ for information on tuning deadtime.
Jul 07 12:48:59 tucan01 heartbeat: [2027]: info: URL: http://linux-ha.org/wiki/FAQ#Heavy_Load
Jul 07 12:48:59 tucan01 heartbeat: [2027]: info: Link tucan02:em2 up.
Jul 07 12:48:59 tucan01 ipfail: [2596]: info: Link Status update: Link tucan02/em2 now has status up
Jul 07 12:48:59 tucan01 heartbeat: [2027]: WARN: Late heartbeat: Node tucan02: interval 200270 ms
Jul 07 12:48:59 tucan01 heartbeat: [2027]: info: Status update for node tucan02: status active
Jul 07 12:48:59 tucan01 ipfail: [2596]: info: Status update: Node tucan02 now has status active
```

Fuente. Autores monografía

6.2 CORTE DEL FLUIDO ELÉCTRICO

En el segundo caso se simuló una falla en el fluido eléctrico, interrumpiendo el suministro de corriente al nodo principal tucan01. Se observaron los logs de HeartBeat para medir el tiempo de respuesta.

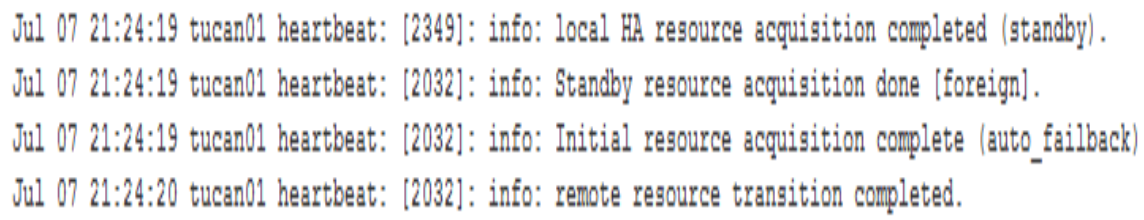
Figura 20. Logs del HeartBeat para medir el tiempo de respuesta

```
Jul 07 21:21:23 tucan02 heartbeat: [1712]: info: Link tucan01:em2 dead.
Jul 07 21:21:23 tucan02 ipfail: [1934]: info: NS: We are dead. :<
harc[3823]:     2011/07/07_21:21:23 info: Running /etc/ha.d//rc.d/status status
Jul 07 21:21:23 tucan02 heartbeat: [3824]: info: No local resources [/usr/share/heartbeat/ResourceManager list
Jul 07 21:21:23 tucan02 ipfail: [1934]: info: Link Status update: Link tucan01/em2 now has status dead
```

Fuente. Autores monografía

La figura anterior muestra la interrupción del fluido eléctrico del servidor tucan01 a las 21:21:23.

Figura 21. Restauración de servicios



Fuente. Autores monografía

La figura 21 muestra el inicio de la restauración del servicio por parte de tucan01 a partir de las 21:24:19, terminado el levantamiento de los mismos a las 21:24:20.

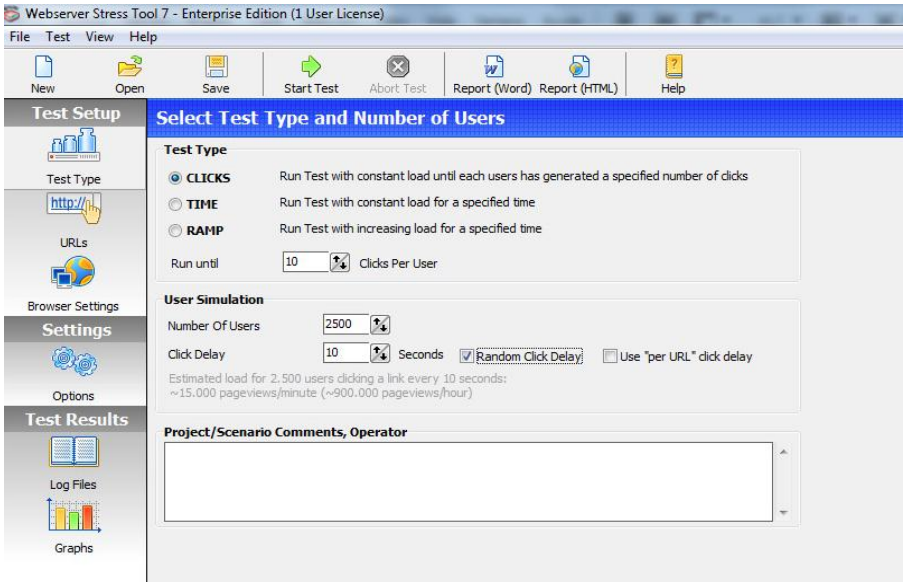
Con base a lo anterior, se corrobora que la restauración de los servicios vuelve a tomar solo 2 min 30 segundos, esto considerando que el sistema cuenta con el uso de un sistema de alimentación ininterrumpida.

6.3 SATURACIÓN DE SERVICIOS

La técnica de saturación de servicios consiste en enviar un gran número de peticiones haciendo que el servidor se sobrecargue y no pueda seguir prestando servicios.[31]

[31] Royer Jean-Marc. Seguridad en la informática de empresa:riesgos, amenazas, prevención y soluciones. Ediciones ENI, 2004 – pag 15-16.

Figura 22. Configuración de Test

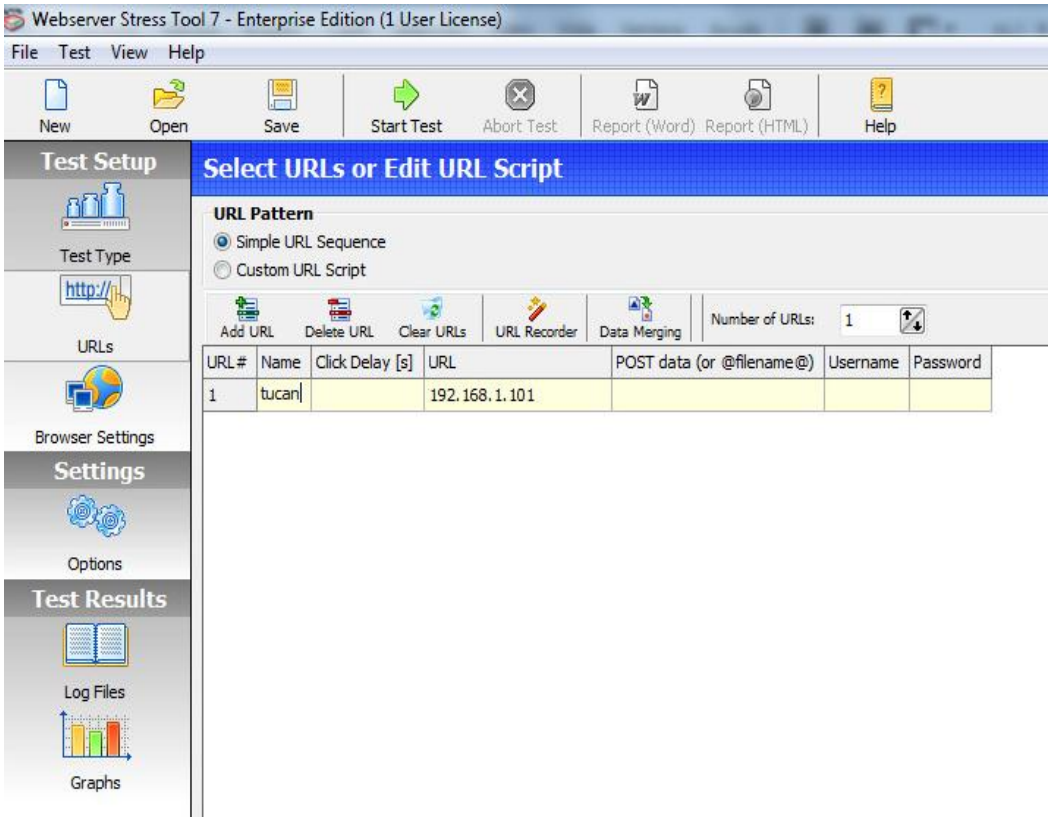


Fuente. Aplicativo Webser Stresss Tool 7

Para la prueba se utilizó una aplicación llamada Webserver Stress Tool, la cual simula a un gran número de usuarios que tienen acceso a un sitio Web. El software puede simular hasta 10.000 usuarios[32], sin embargo se configuro, para que durante 10 minutos, 2500 usuarios enviaran ping a la dirección 192.168.1.101 para saturar el servicio web.

[32] Webser Stress Tool 7 (2011) Disponible en Internet: http://webserver-stress-tool.archivospc.com/

Figura 23. Configuración de la dirección IP



Fuente. Aplicativo Webser Stresss Tool 7

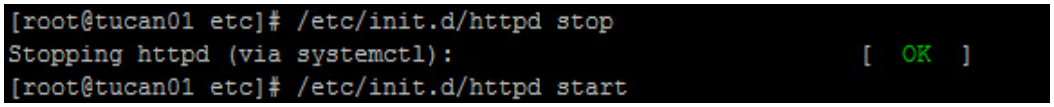
Debido al número de solicitudes enviadas al nodo tucan01, el servicio web colapsó, debido a esta falla el nodo secundario tucan02 toma los servicios (durante un tiempo de 2 min 30 seg).

6.4 ERRORES DE CONFIGURACIÓN

Se producen cuando la persona encargada del soporte del clúster, ejecuta por error un comando, modificándose los archivos de configuración del clúster.

En la siguiente figura muestra que se detiene el servicio httpd del nodo tucan01.

Figura 24. Consola de comandos



Fuente. Autores monografía

En este caso se modificó el nodo principal ocasionando una falla, El nodo tucan02 iniciaría la reactivación de los servicios a los 2 min 30 seg.

Tabla 4. Tiempo de respuesta en fallos

Tiempo respuesta en fallos	
Servidor Dedicado	Servidores Alta disponibilidad
Daños de red (cableado y dispositivos)	
Dependiendo del conocimiento del usuario administrador encargado del soporte, la restauración puede durar desde minutos a horas.	2 min 30 segundos (el nodo respaldo asumiría los servicios automáticamente.
Corte del fluido eléctrico	
La restauración de los servicios puede durar desde unos pocos minutos o más, dependiendo del tiempo que demore en restablecerse el fluido eléctrico.	Con base a las pruebas realizadas se corrobora que la restauración de los servicios vuelve a tomar solo 2 min 30 segundos, además se considera que el sistema cuenta con el uso de un sistema de alimentación ininterrumpida.

Fallas de algún componente Hardware	
Este tipo de servidor al no contar con respaldos físicos al presentarse este tipo de falla, el tiempo de respuesta puede tomar de días a varias semanas, hasta que se adquiera el componente y se configure de nuevo la máquina, esto demora dependerá de modelo del equipo, existencia de la parte afectada, valor por la adquisición, entre otros.	Una de las características en estos servidores es la redundancia de hardware, es decir, estos equipos suelen contar por lo menos con dos dispositivos de apoyo, como routers, switches, servidores, al fallar cualquier componente de alguno de estos, el respaldo automáticamente entraría a suplir el que se dañó.
Saturación del servicio	
El colapso puede ocasionar la interrupción de los servicios durante horas.	2 min (depende si se cuenta con otro proveedor de ISP)
Errores de configuración	
La restauración de los servicios podría demorar desde horas hasta un par de días, esto depende de la prontitud y experiencia del personal técnico en identificar y corregir los errores de configuración.	2 min 30 segundos (el nodo respaldo asumiría los servicios)

Fuente. Autores monografía

CONCLUSIONES

- Este trabajo permitió determinar que los clúster no son solo dispositivos hardware independientes del software que lo integre o su configuración, sino que son componentes de hardware y software interrelacionados de manera física y/o lógica, facilitando la transmisión y actualización de la información entre diferentes servidores de manera simultánea.

- Al trabajar sobre la configuración activo – pasivo, tucan02 envía ping´s de comprobación al nodo principal para verificar si aun esta activo, al no recibir respuesta este nodo reemplaza al servidor principal, iniciando el proceso de recuperación de los servicios en un tiempo aproximado 120 a 150 segundos.

- En el mercado de software libre existen herramientas disponibles para la instalación y configuración de clúster, las cuales son una solución económica alternativa frente al software licenciado.

- La configuración del paquete MySQL en modo activo-activo, asegura la integridad de la información, permitiendo a los dos servidores actualizar y almacenar al tiempo la base de datos. Esto se realiza mediante la instalación del paquete de sincronización Rsync.

- Para visualizar el comportamiento del clúster se pueden instalar paquetes de interfaz gráfica como el Webmin, el cual permite administrar, controlar, modificar y agrega aplicaciones mediante el uso de la interfaz web.

BIBLIOGRAFIA

ARQUÉS M., Huguet, C.M., & Galindo. M. E. (2008). Administración de servidores. En Administración de sistemas operativos en red (pp. 23 - 81) Barcelona: UOC.

BROCHARD J. (2006), Introduction. En J. T. Parra (Ed.), Internet information services (pp.8 - 13).

CASTILLO f (2010). Servicio en red (pp.150-155) España: ed. Nobel.

DUARTE A., & PÉREZ H.M.G. (2006). Sistemas Operativos. En M. Valdez (Ed.), La Informática, Presente Y Futuro En La Sociedad (pp. 121 - 156).Madrid: DYKINSON.

HEURTEL, O.(2009) Introducción a MySQL. En D. Marin (Ed.), PHP y MySQL Domine el desarrollo de un sitio Web Dinámico e interactivo (pp. 15 - 43) Barcelona : ENI

MARTINEZ J. et al. (2009). Servicio. En IPv6 para Todos: Guía de uso y aplicación para diversos entornos (pp.77-78) Argentina: Internet Society.

PERALES MARTÍNEZ David. (2009). Sistemas De Alta Disponibilidad. En Lulu.com (Ed.), *UNIX A BASE DE EJEMPLOS* (pp. 229 - 234) España.

STERLING T. L. (2002). An Overview Of Cluster Computing. En G. Bell (Ed.), Beowulf cluster computing with Linux (pp.15 - 29). MIT Press

THIBAUD C. (2006). Presentación de MySQL. En MySQL 5 (p.p.6) Barcelona :ENI

WEYGANT P. S. (2001). Basic High Availability. EN D. Cullen-Dolce (Ed.),Cluster for high Availability, Prentice Hall (pp. 1 - 38) E.E.U.U.: Prentice-Hall, Inc.

Documentación disponible en Internet:

Single point of failure, (2011, junio).
En http://es.wikipedia.org/wiki/Single_point_of_failure

Fedora (2011) recuperado el 06 de junio de 2011 del sitio web: http://fedoraproject.org/es/

GNOME 3 made of easy (2011) recuperado el 06 de junio de 2011 del sitio web: http://gnome3.org/index.html.es

Bind (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.isc.org/software/bind/whatis

Lackerbauer I. (2009). La Descarga De Un Archivo En L. Fontana, (Trad.). Internet Todo Sobre Internet.(pp. 125 -13). España: BOIXAREU(Trabajo original publicado en 2000).

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/doc/users-guide/_heartbeat_as_a_cluster_messaging_layer.html

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/doc/users-guide/ _components.html

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/doc/users-guide/ _cluster_consensus_membership.html

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/doc/users-guide/ _cluster_plumbing_library.html

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/doc/users-guide/ _ipc_library.html

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/doc/users-guide/ _non_blocking_logging_daemon.html

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/doc/man-pages/re-authkeys.html

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/ doc/man-pages/re-hacf.html

Heartbeat (2011) recuperado el 06 de junio de 2011 del sitio web: http://www.linux-ha.org/doc/man-pages/re-hacf.html

What is the Linux Virtual Server (20011) recuperado el 09 de Julio 2011 del sitio web http://www.linuxvirtualserver.or

ANEXOS

Anexo 1. Instalación de Sistema Operativo Fedora 15

El siguiente procedimiento muestra la instalación básica de Fedora, una distribución de Linux elaborada con software libre y de código abierto. El proceso ilustrado ayudará a instalar Fedora para el montaje de sistema de alta disponibilidad.

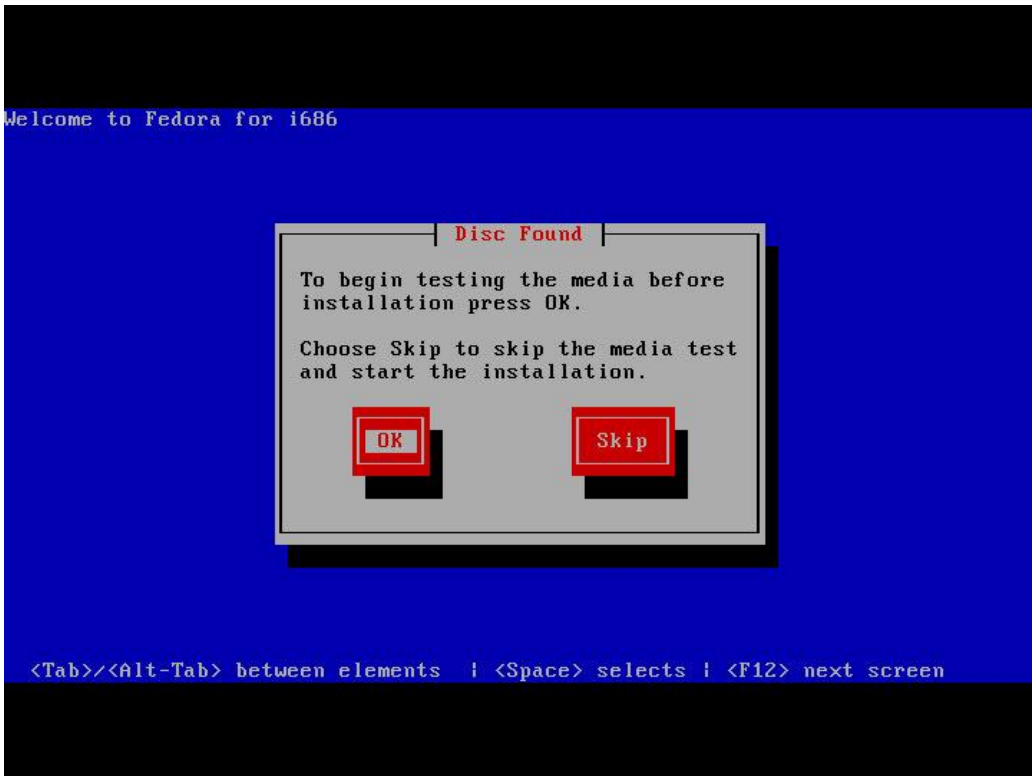
El proceso de instalación es simple y se realiza en pocos pasos:

1. Descargar los archivos para construir el medio de arranque
2. Arrancar el equipo e iniciar el proceso de instalación
3. Reiniciar y realizar la configuración post-instalación.

La descarga de los archivos para construir el medio de arranque se realizó desde el sitio web oficial del proyecto Fedora en su versión en español www.fedoraproject.org/es/ . En el momento de la elaboración de esta guía la última versión estable disponible para arquitectura i686 era la versión 15 Live Desktop.

Con el instalador de Fedora 15 en la unidad de DVD del equipo, se procede a iniciar el arranque desde esta unidad y aparece la primera pantalla del proceso de instalación, a partir de este momento, el proceso se enumerará por pasos:

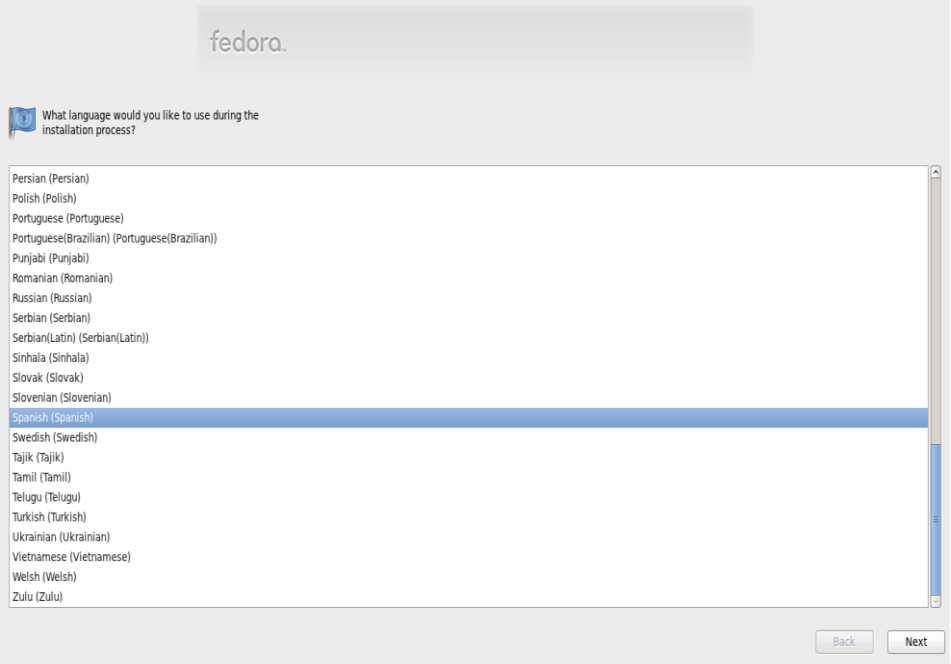
Figura 25: Inicio Instalación Fedora



Fuente. Instalador Fedora

Paso 3: Se debe seleccionar el idioma en el cual se realizará la instalación, para este caso Spanish (Español) y se da clic en Next.

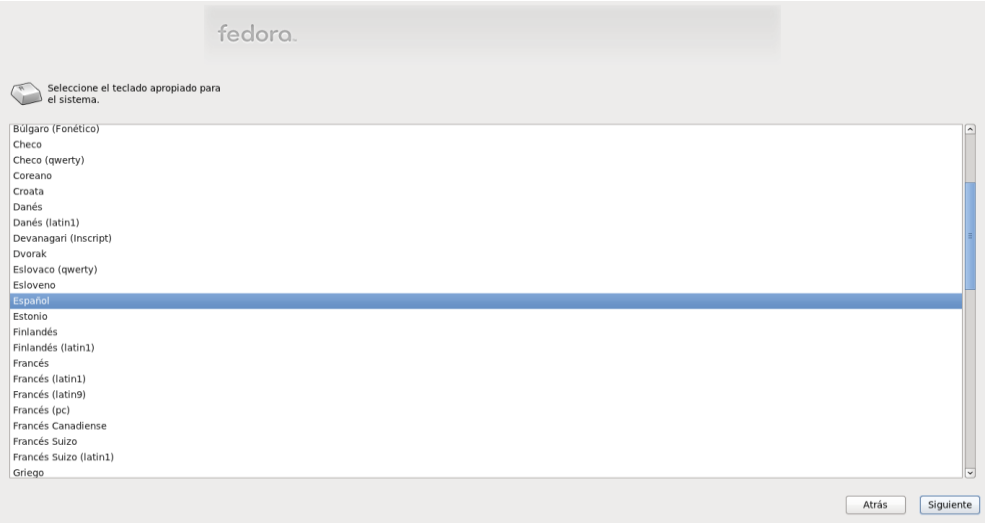
Figura 26: Instalación Fedora configuración del idioma



Fuente. Instalador Fedora

Paso 4: Seleccionar el teclado apropiado para el sistema, desde este punto los mensajes de la instalación aparecen en idioma español, el cual fue seleccionado en el paso anterior. La distribución del teclado usada en la instalación español. Clic en siguiente.

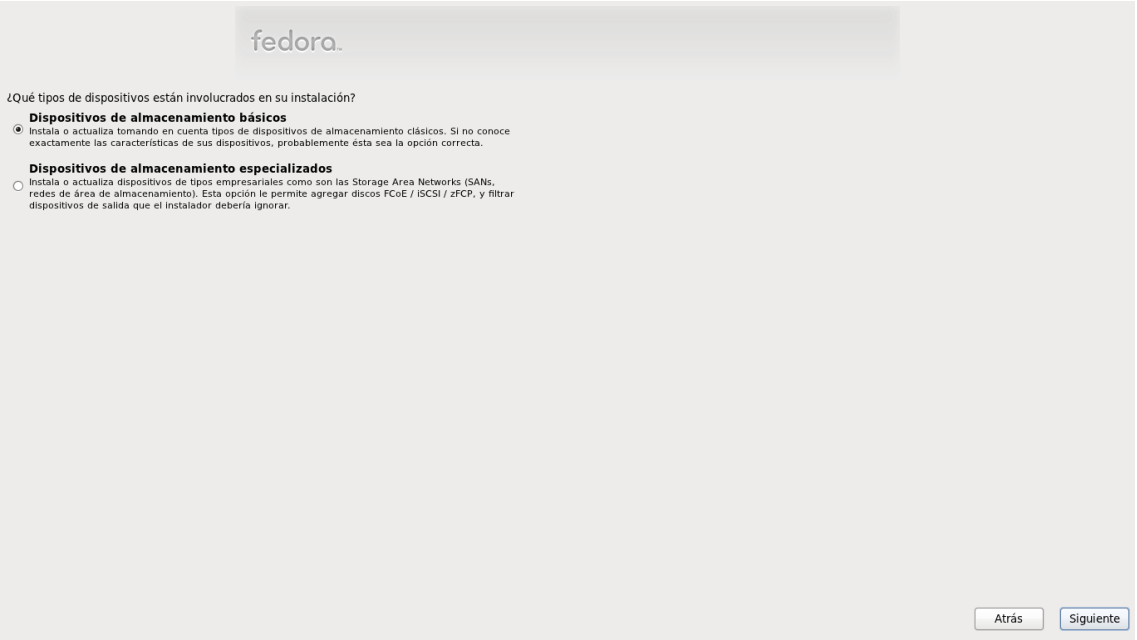
Figura 27: Fedora 15 configuración de teclado



Fuente. Instalador Fedora

Paso 5: Selección del tipo de dispositivos involucrados en la instalación. El equipo donde se está realizando la instalación cuenta con un dispositivo básico (Disco Duro Sata) la opción a elegir es **Dispositivos de almacenamiento básicos**. Clic en Siguiente.

Figura 28: Fedora 15 selección de dispositivo almacenamiento



Fuente. Instalador Fedora

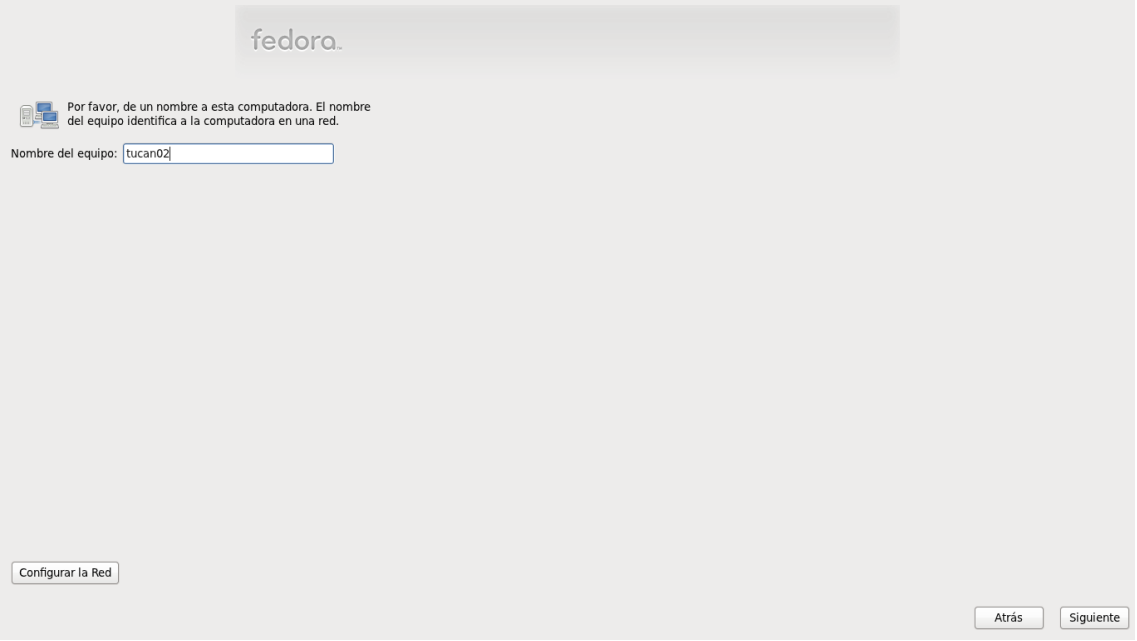
Paso 6: El proceso de instalación advierte que el dispositivo de almacenamiento podría contener datos. Para esta instalación se está usando un disco duro nuevo, por lo cual se escoge la opción Si, eliminar cualquier tipo de datos. Clic en Siguiente.

Figura 29: Fedora 15 aceptar el dispositivo de almacenamiento

Fuente. Instalador Fedora

Paso 6: Se debe indicar el nombre del equipo, para este caso Tucan02. El nombre corresponde al que haya sido designado para el servidor que se está instalando. Clic en siguiente.

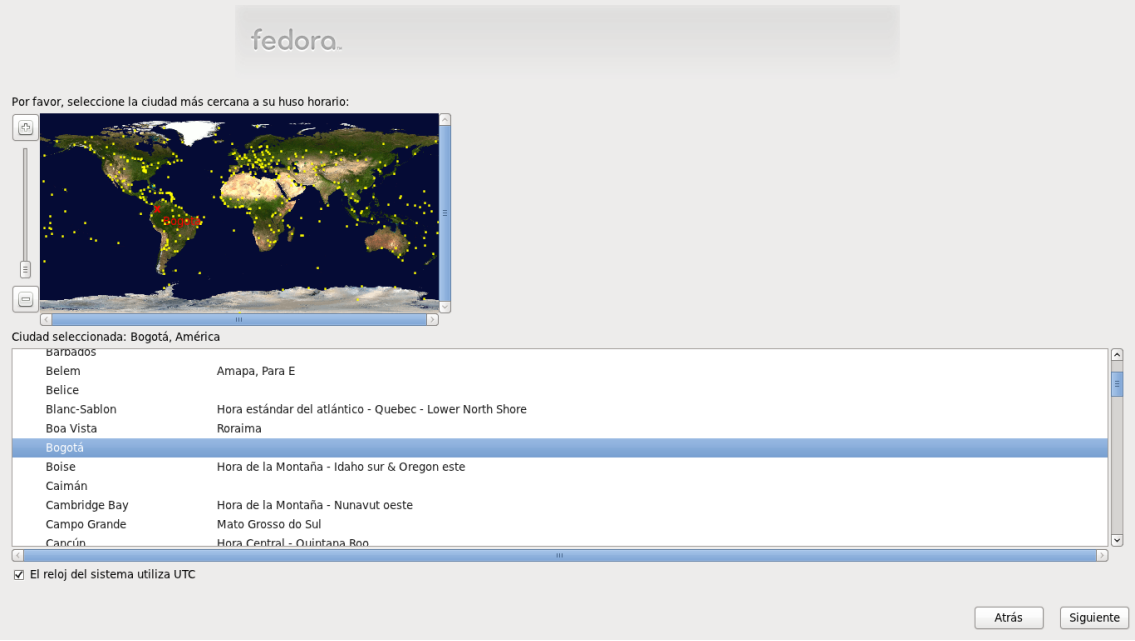
Figura 30: Fedora 15 asignación de usuario



Fuente. Instalador Fedora

Paso 7: Seleccionar la ciudad más cercana al huso horario que tendrá el servidor y clic en siguiente.

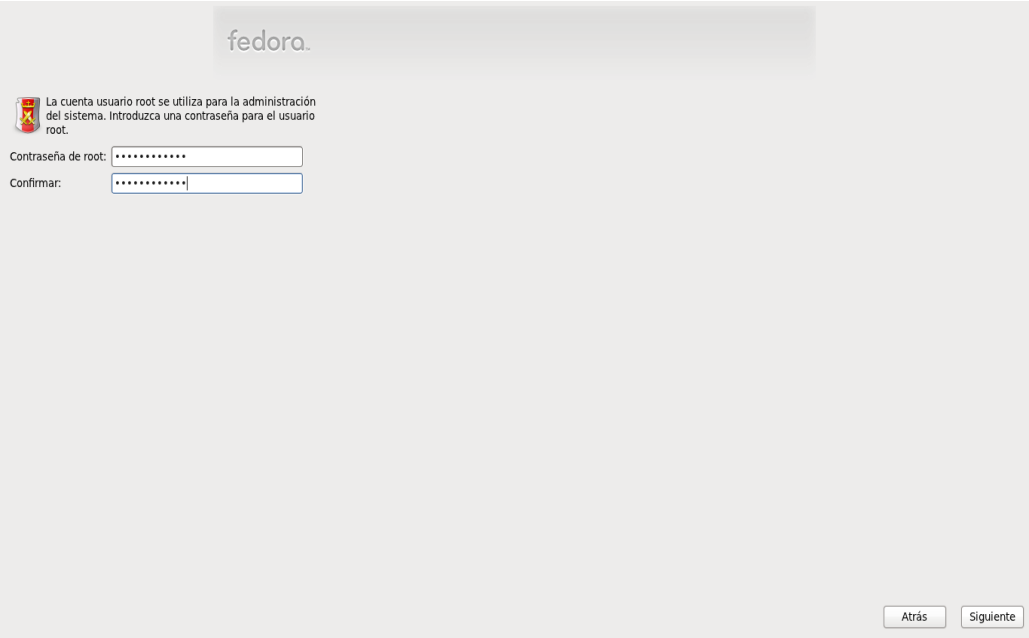
Figura 31: Fedora 15 zona horaria



Fuente. Instalador Fedora

Paso 8: Indicar la contraseña del usuario root. Clic en siguiente.

Figura 32: Fedora 15 usuario y contraseña



Fuente. Instalador Fedora

Paso 9. El proceso de instalación muestra diferentes opciones para el tipo de instalación que se desea.

Utilizar todo el disco: Elimina todas las particiones en el disco duro y crea una estructura automáticamente.

Reemplazar sistemas Linux Existentes Elimina las particiones Linux que encuentre como Ext3, Ext4, e instala fedora remplazando las particiones.

Achicar el sistema actual modifica las particiones y crea la estructura para instalación.

Utilizar el espacio libre se instala en el espacio libre no particionado y crea las particiones que necesite.

Crear un diseño personalizado me permite crear una estructura de particiones personalizadas.

En la instalación que se realiza se escoge **Utilizar todo el espacio**. Clic en siguiente.

Figura 33: Fedora 15 partición de disco

Fuente. Instalador Fedora

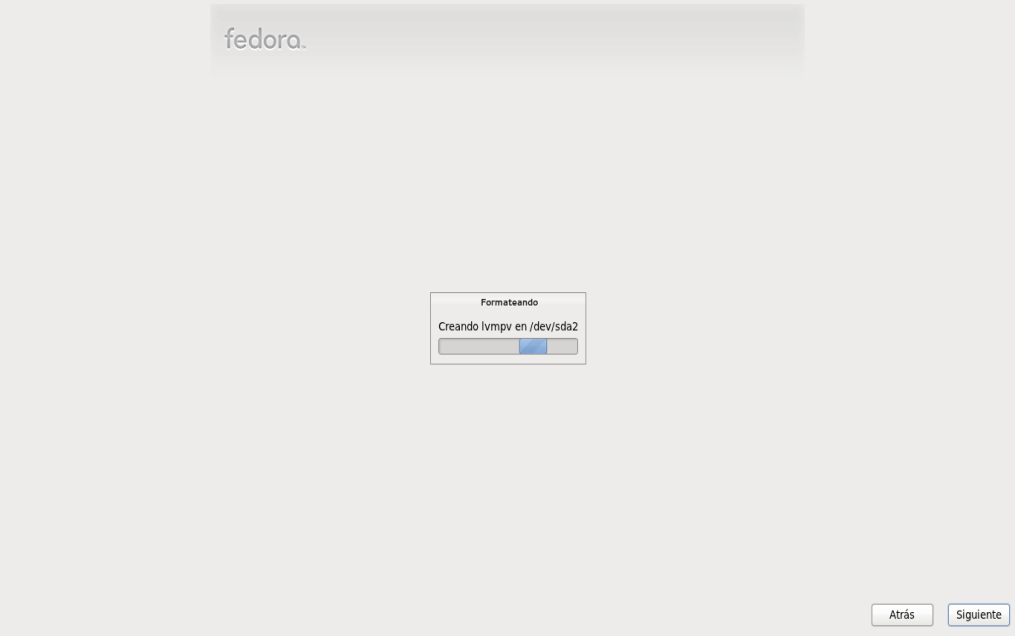
Paso 10: En este punto de la instalación se realizarán cambios que ya no se podrán deshacer. Se puede retroceder para modificar alguna de las opciones en los pasos anteriores o Guardar los cambios al disco. Si todo esta correcto se selecciona Guardar cambios al disco. Clic en Siguiente.

Figura 34: Fedora 15 inicia instalación

Fuente. Instalador Fedora

Paso 11: La instalación de Fedora muestra el proceso de formateado del disco. Clic en Siguiente.

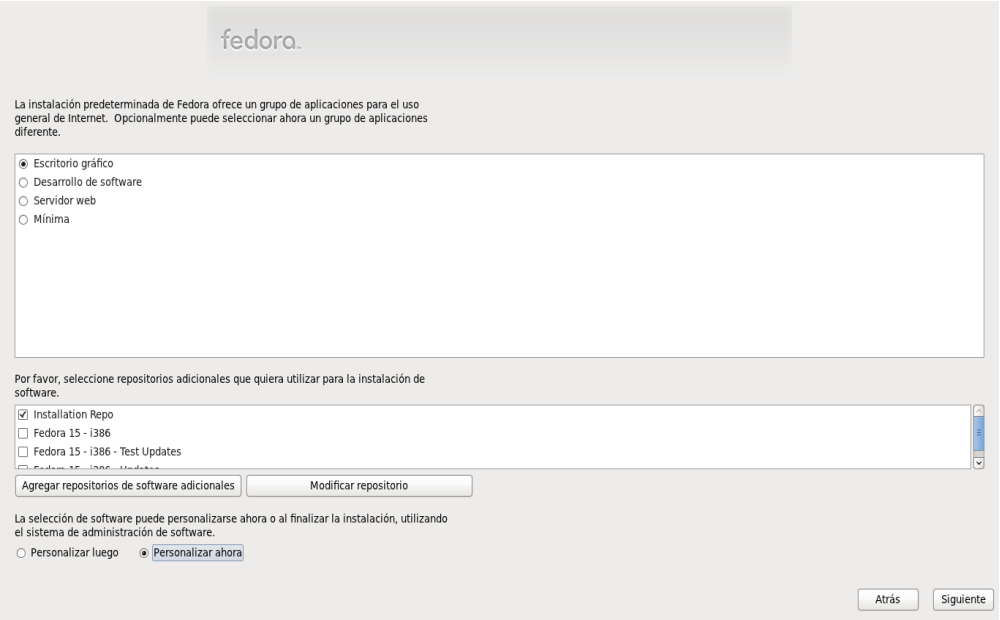
Figura 35: Fedora 15 copiando archivos



Fuente. Instalador Fedora

Paso 12: La instalación ofrece un grupo de aplicaciones para el uso general de internet. A pesar de que esta instalación es para un servidor de alta disponibilidad, se ha escogido montar el escritorio gráfico como una herramienta de soporte al acceder al servidor. También se ha seleccionado que se instalen repositorios y la opción de personalizar cuales repositorios se instalara. Clic en Siguiente.

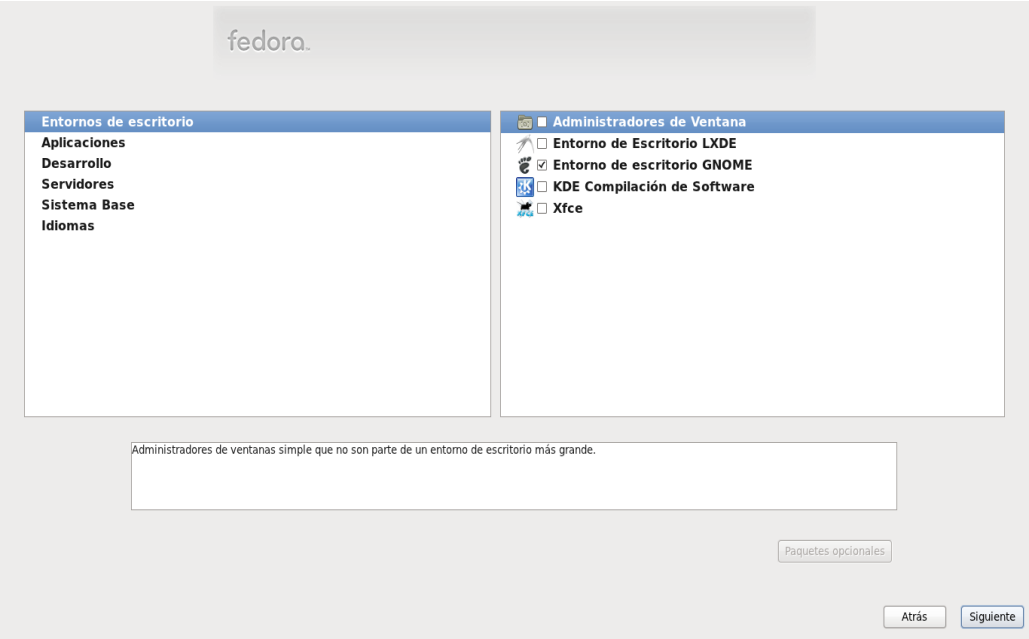
Figura 36: Fedora 15 escritorio



Fuente. Instalador Fedora

Paso 13: Configurando repositorios. En la opción Entorno de escritorio se ha elegido el entorno de escritorio GNOME.

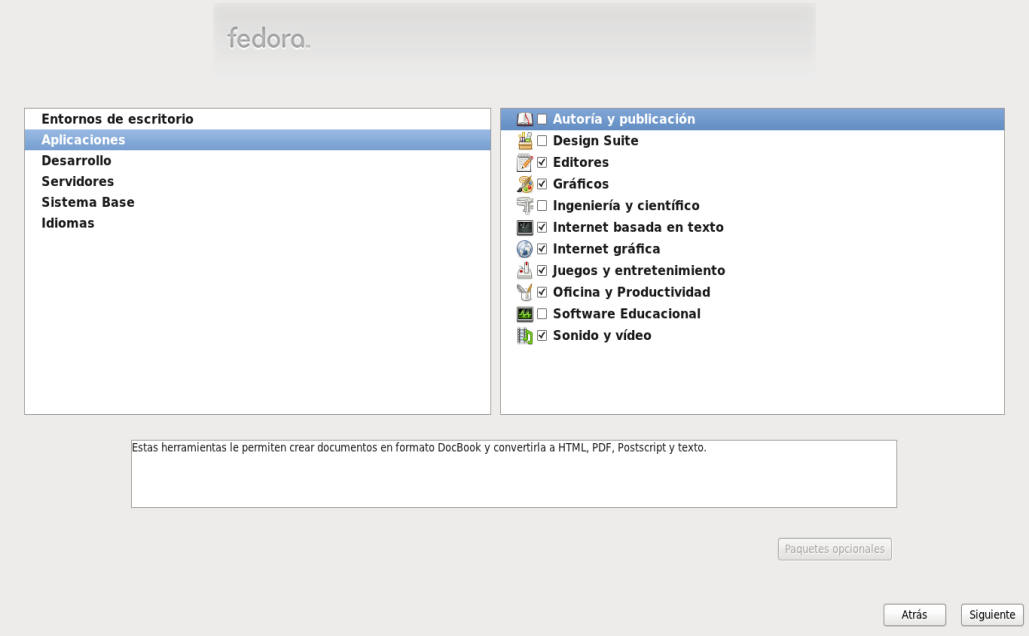
Figura 37: Fedora 15 selección de escritorio GNOME



Fuente. Instalador Fedora

En la opción Aplicaciones se ha dejado las opciones seleccionadas por defecto.

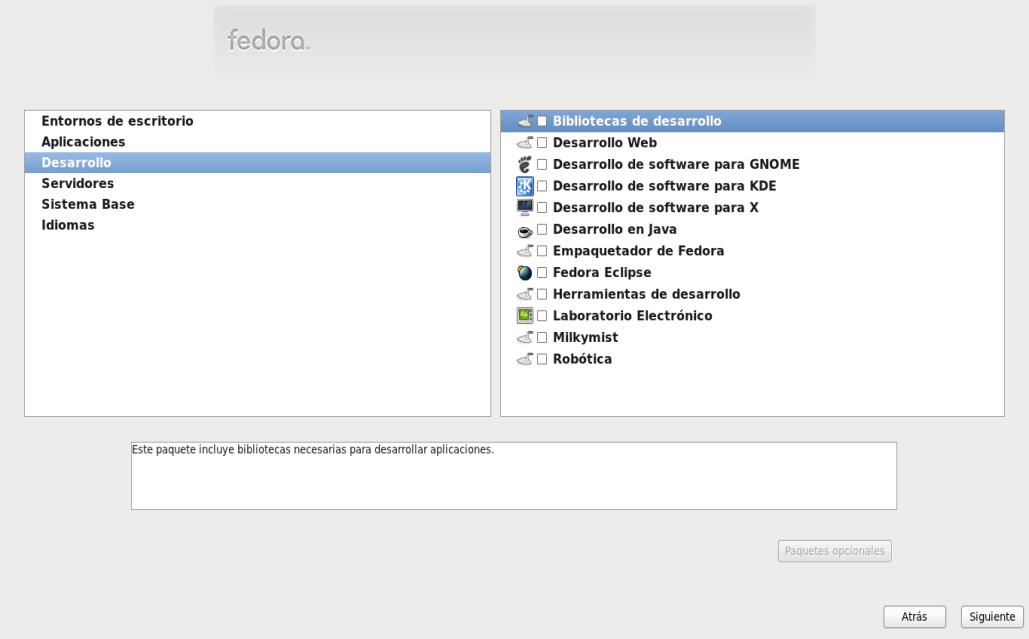
Figura 38: Fedora 15 instalación de aplicaciones



Fuente. Instalador Fedora

En la opción Desarrollo se han deshabilitado todas las opciones.

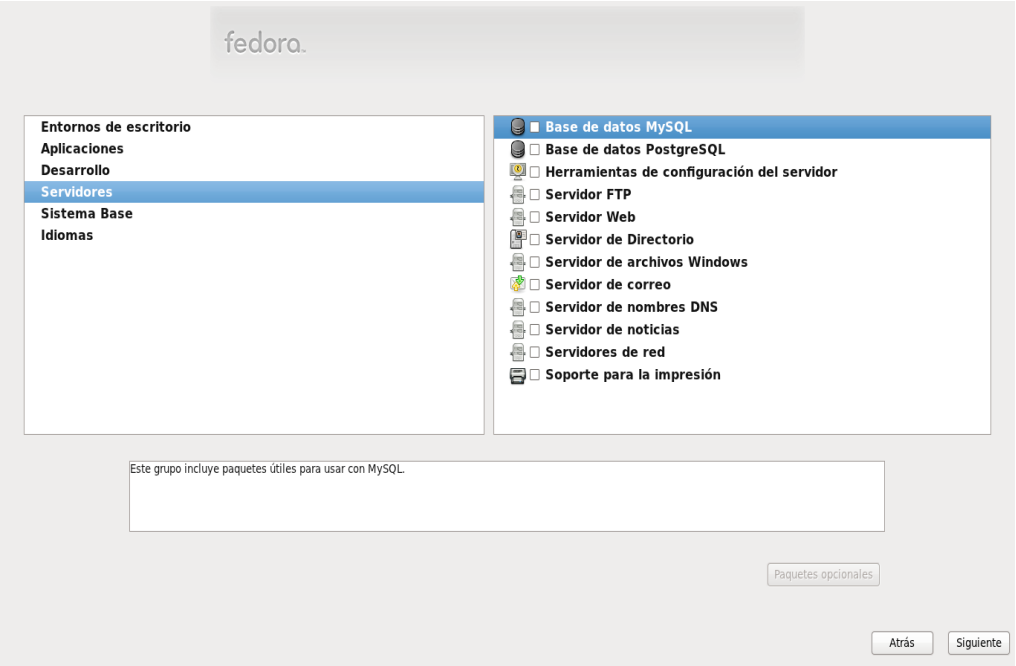
Figura 39: Fedora 15 aplicaciones de desarrollo



Fuente. Instalador Fedora

En la opción Servidores se han deshabilitado todas las opciones, ya que los servicios a utilizar serán instalados y configurados más adelante.

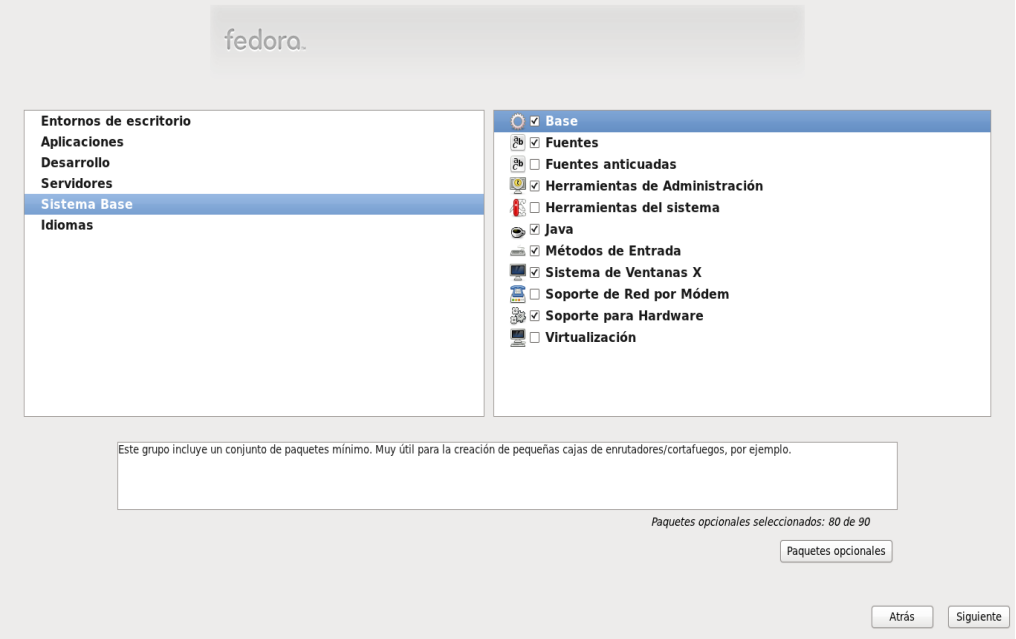
Figura 40: Fedora 15 servidores



Fuente. Instalador Fedora

En la opción Sistema Base se han dejado seleccionadas las opciones por defecto.

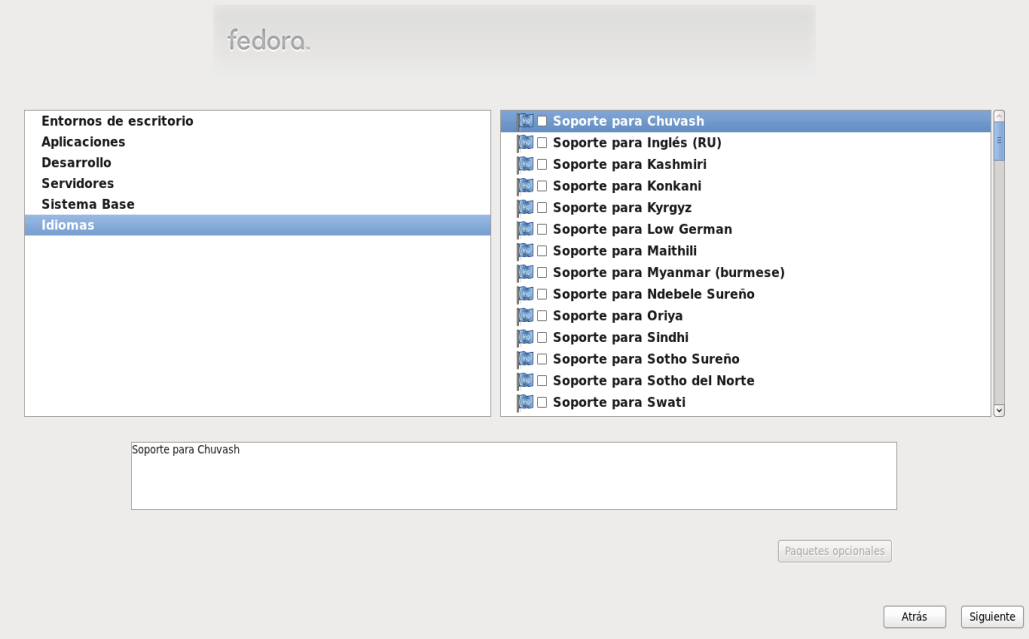
Figura 41: Fedora 15 más complementos



Fuente. Instalador Fedora

En la opción Idiomas se ha dejado la selección por defecto. Clic en Siguiente.

Figura 42: Fedora 15 paquete de idiomas



Fuente. Instalador Fedora

Imagen del proceso de instalación.

Figura 43: Fedora 15 instalación

Fuente. Instalador Fedora

Paso 14: Una vez terminado el proceso de instalación se muestra el mensaje donde se indica que se debe reiniciar el equipo para poder utilizar el nuevo sistema instalado. Clic en Reiniciar.

Figura 44: Fedora 15 fin de la instalación

Fuente. Instalador Fedora

Una vez terminada la instalación, el sistema solicita que se reinicie el equipo.

Anexo 2. Instalación de Webmin

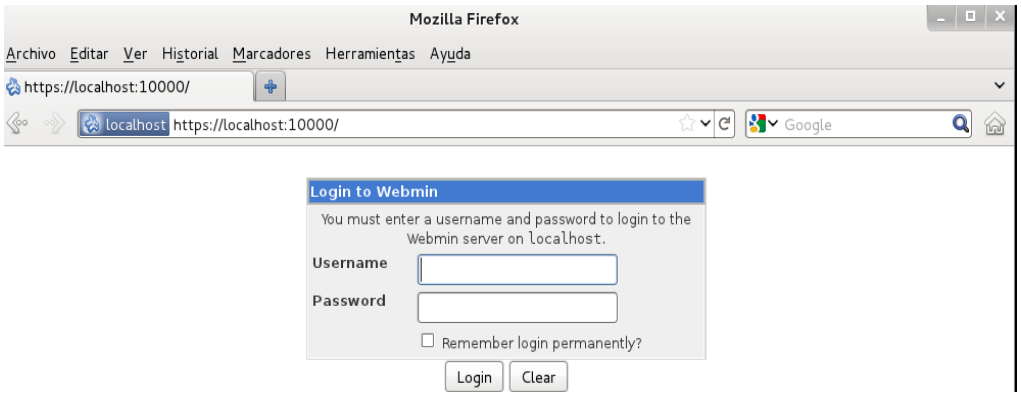
Se descarga el paquete de instalación de Webmin desde la consola con el siguiente comando:

wget http:://prdownloads.sourceforge.net/webadmin/webmin-1.550-1.noarch.rpm.

A continuación se compila el paquete de instalación ejecutando el comando: rpm –i webmin-1.550-1.noarch.rpm

Una vez terminada la instalación de Webmin, se puede ingresar al URL https://localhost:10000/ donde se muestra la ventana de logueo. Se accede con el usuario root y su contraseña.

Figura 45: Ingreso Webmin



Fuente. Aplicativo Webmin

Anexo 3. Configuración Authkeys

auth 1

1 crc

3 md5

2 sha1

Anexo 4. Archivo de configuración heartbeat Ha

```
#
#       There are lots of options in this file.  All you have to have is a set
#       of nodes listed {"node ...} one of {serial, bcast, mcast, or ucast},
#       and a value for "auto_failback".
#
#       ATTENTION: As the configuration file is read line by line,
#        THE ORDER OF DIRECTIVE MATTERS!
#
#       In particular, make sure that the udpport, serial baud rate
#       etc. are set before the heartbeat media are defined!
#       debug and log file directives go into effect when they
#       are encountered.
#
#       All will be fine if you keep them ordered as in this example.
#
#
#       Note on logging:
#       If all of debugfile, logfile and logfacility are not defined,
#       logging is the same as use_logd yes. In other case, they are
#       respectively effective. if detering the logging to syslog,
#       logfacility must be "none".
#
#       File to write debug messages to
debugfile /var/log/ha-debug
#
```

```
#
#       File to write other messages to
#
logfile /var/log/ha-log
#
#
#       Facility to use for syslog()/logger
#
logfacility     local0
#
#
#       A note on specifying "how long" times below...
#
#       The default time unit is seconds
#       10 means ten seconds
#
#       You can also specify them in milliseconds
#       1500ms means 1.5 seconds
#
#
#       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 30
#
#       warntime: how long before issuing "late heartbeat" warning?
#       See the FAQ for how to use warntime to tune deadtime.
#
warntime 10
#
#
#       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
#
#       Baud rate for serial ports...
#
#baud  19200
#
#       serial    serialportname ...
```

```
#serial  /dev/ttyS0   # Linux
#serial  /dev/cuaa0 # FreeBSD
#serial /dev/cuad0     # FreeBSD 6.x
#serial/dev/cua/a    # Solaris
#
#
#      What interfaces to broadcast heartbeats over?
#
bcast  em2            # Linux
#bcasteth1 eth2      # Linux
#bcastle0             # Solaris
#bcastle1 le2         # Solaris
#
#      Set up a multicast heartbeat medium
#      mcast [dev] [mcast group] [port] [ttl] [loop]
#
#      [dev]            device to send/rcv heartbeats on
#      [mcast group]  multicast group to join (class D multicast address
#                224.0.0.0 - 239.255.255.255)
#      [port]           udp port to sendto/rcvfrom (set this value to the
#                same value as "udpport" above)
#      [ttl]            the ttl value for outbound heartbeats.  this effects
#                how far the multicast packet will propagate.  (0-255)
#                Must be greater than zero.
#      [loop]           toggles loopback for outbound multicast heartbeats.
#                if enabled, an outbound packet will be looped back and
#                received by the interface it was sent on. (0 or 1)
#                Set this value to zero.
#
```

```
#
#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 em2 192.168.1.129
#
#
#       About boolean values...
#
#       Any of the following case-insensitive values will work for true:
#         true, on, yes, y, 1
#       Any of the following case-insensitive values will work for false:
#         false, off, no, n, 0
#
#
#
#       auto_failback:  determines whether a resource will
#       automatically fail back to its "primary" node, or remain
#       on whatever node is serving it until that node fails, or
#       an administrator intervenes.
#
#       The possible values for auto_failback are:
#         on      - enable automatic failbacks
#         off     - disable automatic failbacks
```

```
#       legacy- enable automatic failbacks in systems
#               where all nodes do not yet support
#               the auto_failback option.
#
#      auto_failback "on" and "off" are backwards compatible with the old
#       "nice_failback on" setting.
#
#      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
#
#
#       Basic STONITH support
#       Using this directive assumes that there is one stonith
#       device in the cluster.  Parameters to this device are
#       read from a configuration file. The format of this line is:
#
#         stonith <stonith_type> <configfile>
#
#       NOTE: it is up to you to maintain this file on each node in the
#       cluster!
#
```

```
#stonith baytech /etc/ha.d/conf/stonith.baytech
#
#       STONITH support
#       You can configure multiple stonith devices using this directive.
#       The format of the line is:
#         stonith_host <hostfrom> <stonith_type> <params...>
#         <hostfrom> is the machine the stonith device is attached
#             to or * to mean it is accessible from any host.
#         <stonith_type> is the type of stonith device (a list of
#             supported drives is in /usr/lib/stonith.)
#         <params...> are driver specific parameters.  To see the
#             format for a particular device, run:
#           stonith -l -t <stonith_type>
#
#
#       Note that if you put your stonith device access information in
#       here, and you make this file publically readable, you're asking
#       for a denial of service attack ;-)
#
#       To get a list of supported stonith devices, run
#         stonith -L
#       For detailed information on which stonith devices are supported
#       and their detailed configuration options, run this command:
#         stonith -h
#
#stonith_host *      baytech 10.0.0.3 mylogin mysecretpassword
#stonith_host ken3  rps10 /dev/ttyS1 kathy 0
#stonith_host kathy rps10 /dev/ttyS1 ken3 0
#
```

```
#       Watchdog is the watchdog timer.  If our own heart doesn't beat for
#       a minute, then our machine will reboot.
#       NOTE: If you are using the software watchdog, you very likely
#       wish to load the module with the parameter "nowayout=0" or
#       compile it without CONFIG_WATCHDOG_NOWAYOUT set.
Otherwise even
#       an orderly shutdown of heartbeat will trigger a reboot, which is
#       very likely NOT what you want.
#
#watchdog /dev/watchdog
#
#       Tell what machines are in the cluster
#       node    nodename ...-- must match uname -n
node    tucan01
node tucan02
#
#       Less common options...
#
#       Treats 10.10.10.254 as a psuedo-cluster-member
#       Used together with ipfail below...
#       note: don't use a cluster node as ping node
#
#ping 10.10.10.254
#
#       Treats 10.10.10.254 and 10.10.10.253 as a psuedo-cluster-
member
#       called group1. If either 10.10.10.254 or 10.10.10.253 are up
#       then group1 is up
#       Used together with ipfail below...
```

```
#
#ping_group group1 10.10.10.254 10.10.10.253
#
#       HBA ping derective for Fiber Channel
#       Treats fc-card-name as psudo-cluster-member
#       used with ipfail below ...
#
#       You can obtain HBAAPI from http://hbaapi.sourceforge.net.  You
need
#       to get the library specific to your HBA directly from the vender
#       To install HBAAPI stuff, all You need to do is to compile the
common
#       part you obtained from the sourceforge. This will produce
libHBAAPI.so
#       which you need to copy to /usr/lib. You need also copy hbaapi.h to
#       /usr/include.
#
#       The fc-card-name is the name obtained from the hbaapitest
program
#       that is part of the hbaapi package. Running hbaapitest will produce
#       a verbose output. One of the first line is similar to:
#         Apapter number 0 is named: qlogic-qla2200-0
#       Here fc-card-name is qlogic-qla2200-0.
#
#hbaping fc-card-name
#
#
#       Processes started and stopped with heartbeat.  Restarted unless
#         they exit with rc=100
```

```
#

#respawn userid /path/name/to/run
respawn hacluster /usr/lib/heartbeat/ipfail
#
#                                               Access control for client
api
#                                               default is no access
#
#apiauth client-name gid=gidlist uid=uidlist
#apiauth ipfail gid=haclient uid=hacluster

##############################
#
#                                               Unusual options.
#
##############################
#
#       hopfudge maximum hop count minus number of nodes in config
#hopfudge 1
#
#       deadping - dead time for ping nodes
#deadping 30
#
#       hbgenmethod - Heartbeat generation number creation method
#         Normally these are stored on disk and incremented as needed.
#hbgenmethod time
#
#       realtime - enable/disable realtime execution (high priority, etc.)
```

```
#       defaults to on
#realtime off
#
#       debug - set debug level
#       defaults to zero
#debug 1
#
#       API Authentication - replaces the fifo-permissions-based system of
the past
#
#
#       You can put a uid list and/or a gid list.
#       If you put both, then a process is authorized if it qualifies under
either
#       the uid list, or under the gid list.
#
#       The groupname "default" has special meaning.  If it is specified,
then
#       this will be used for authorizing groupless clients, and any client
groups
#       not otherwise specified.
#
#       There is a subtle exception to this.  "default" will never be used in
the
#       following cases (actual default auth directives noted in brackets)
#          ipfail (uid=HA_CCMUSER)
#          ccm            (uid=HA_CCMUSER)
#          ping           (gid=HA_APIGROUP)
#          cl_status      (gid=HA_APIGROUP)
```

```
#
#       This is done to avoid creating a gaping security hole and matches
the most
#       likely desired configuration.
#
#apiauth ipfail uid=hacluster
#apiauth ccm uid=hacluster
#apiauth cms uid=hacluster
#apiauth ping gid=haclient uid=alanr,root
#apiauth default gid=haclient

#       message format in the wire, it can be classic or netstring,
#       default: classic
#msgfmt  classic/netstring

#       Do we use logging daemon?
#       If logging daemon is used, logfile/debugfile/logfacility in this file
#       are not meaningful any longer. You should check the config file for
logging
#       daemon (the default is /etc/logd.cf)
#       more infomartion can be fould in the man page.
#       Setting use_logd to "yes" is recommended
#
use_logd yes
#
#       the interval we  reconnect to logging daemon if the previous
connection failed
#       default: 60 seconds
#conn_logd_time 60
```

```
#
#       Configure compression module
#       It could be zlib or bz2, depending on whether u have the
corresponding
#       library   in the system.
#compression         bz2
#
#       Confiugre compression threshold
#       This value determines the threshold to compress a message,
#       e.g. if the threshold is 1, then any message with size greater than 1
KB
#       will be compressed, the default is 2 (KB)
#compression_threshold 2
```

Anexo 5. Configuración de archivo haresources

tucan01 IPaddr2::192.168.1.10/24/em2 httpd MySQLd named

#

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.

#

<VERY IMPORTANT NOTE>

The haresources files MUST BE IDENTICAL on all nodes of the cluster.

#

The node names listed in front of the resource group information

is the name of the preferred node to run the service. It is

not necessarily the name of the current machine. If you are running

auto_failback ON (or legacy), then these services will be started

up on the preferred nodes - any time they're up.

#

If you are running with auto_failback OFF, then the node information

```
#       will be used in the case of a simultaneous start-up, or when using
#       the hb_standby {foreign,local} command.
#
#       BUT FOR ALL OF THESE CASES, the haresources files MUST BE IDENTICAL.
#       If your files are different then almost certainly something
#       won't work right.
# </VERY IMPORTANT NOTE>
#
#
#       We refer to this file when we're coming up, and when a machine is being
#       taken over after going down.
#
#       You need to make this right for your installation, then install it in
#       /etc/ha.d
#
#       Each logical line in the file constitutes a "resource group".
#       A resource group is a list of resources which move together from
#       one node to another - in the order listed.  It is assumed that there
```

```
#       is no relationship between different resource groups.  These
#       resource in a resource group are started left-to-right, and stopped
#       right-to-left.  Long lists of resources can be continued from line
#       to line by ending the lines with backslashes ("\").
#
#       These resources in this file are either IP addresses, or the name
#       of scripts to run to "start" or "stop" the given resource.
#
#       The format is like this:
#
#node-name resource1 resource2 ... resourceN
#
#
#       If the resource name contains an :: in the middle of it, the
#       part after the :: is passed to the resource script as an argument.
#       Multiple arguments are separated by the :: delimeter
#
#       In the case of IP addresses, the resource script name IPaddr is
#       implied.
```

```
#
#       For example, the IP address 135.9.8.7 could also be represented
#       as IPaddr::135.9.8.7
#
#       THIS IS IMPORTANT!!
#       The given IP address is directed to an interface which has a route
#       to the given address.  This means you have to have a net route
#       set up outside of the High-Availability structure.  We don't set it
#       up here -- we key off of it.
#
#       The broadcast address for the IP alias that is created to support
#       an IP address defaults to the highest address on the subnet.
#
#       The netmask for the IP alias that is created defaults to the same
#       netmask as the route that it selected in in the step above.
#
#       The base interface for the IPalias that is created defaults to the
#       same netmask as the route that it selected in in the step above.
#
```

```
#       If you want to specify that this IP address is to be brought up
#       on a subnet with a netmask of 255.255.255.0, you would specify
#       this as IPaddr::135.9.8.7/24 .
#
#       If you wished to tell it that the broadcast address for this subnet
#       was 135.9.8.210, then you would specify that this way:
#         IPaddr::135.9.8.7/24/135.9.8.210
#
#       If you wished to tell it that the interface to add the address to
#       is eth0, then you would need to specify it this way:
#         IPaddr::135.9.8.7/24/eth0
#
#       And this way to specify both the broadcast address and the
#       interface:
#         IPaddr::135.9.8.7/24/eth0/135.9.8.210
#
#       The IP addresses you list in this file are called "service" addresses,
#       since they're they're the publicly advertised addresses that clients
#       use to get at highly available services.
```

```
#
#       For a hot/standby (non load-sharing) 2-node system with only
#       a single service address,
#       you will probably only put one system name and one IP address in
here.
#       The name you give the address to is the name of the default "hot"
#       system.
#
#       Where the nodename is the name of the node which "normally"
owns the
#       resource.  If this machine is up, it will always have the resource
#       it is shown as owning.
#
#       The string you put in for nodename must match the uname -n
name
#       of your machine.  Depending on how you have it administered, it
could
#       be a short name or a FQDN.
#
#-------------------------------------------------------------------
#
```

```
#       Simple case: One service address, default subnet and netmask
#         No servers that go up and down with the IP address
#
#just.linux-ha.org     135.9.216.110
#
#---------------------------------------------------------------
#
#       Assuming the adminstrative addresses are on the same subnet...
#       A little more complex case: One service address, default subnet
#       and netmask, and you want to start and stop http when you get
#       the IP address...
#
#just.linux-ha.org     135.9.216.110 http
#---------------------------------------------------------------
#
#       A little more complex case: Three service addresses, default
subnet
#       and netmask, and you want to start and stop http when you get
#       the IP address...
#
```

```
#just.linux-ha.org     135.9.216.110 135.9.215.111 135.9.216.112 httpd
#---------------------------------------------------------------------
#
#       One service address, with the subnet, interface and bcast addr
#       explicitly defined.
#
#just.linux-ha.org     135.9.216.3/28/eth0/135.9.216.12 httpd
#
#---------------------------------------------------------------------
#
#       An example where a shared filesystem is to be used.
#       Note that multiple aguments are passed to this script using
#       the delimiter '::' to separate each argument.
#
#node1  10.0.0.170 Filesystem::/dev/sda1::/data1::ext2
#
#       Regarding the node-names in this file:
#
#       They must match the names of the nodes listed in ha.cf, which in
turn
```

must match the `uname -n` of some node in the cluster. So they aren't

virtual in any sense of the word.

#

g/

GLOSARIO

Apache (Acrónimo de "A patchy server"). Es un servidor web de distribución libre y código abierto.

Broadcast. Modo de transmisión de información desde un nodo emisor a todos los dispositivos de una red.

CRC (Cyclic Redundacy check). Es una técnica utilizada para la detección de errores, funciona protegiendo la información del mensaje mediante la adicción reiterada de bits de chequeo al final de la transmisión.

Cron. Programa que permite a usuarios Linux ejecutar automáticamente comandos en una fecha específica.

DAEMON (Disk And Execution Monitor). Es un proceso informático no interactivo que se ejecuta en forma continua sin intervención del usuario.

EXT3 (Third Extended Filesystem) – Es un sistema de archivos con registro por diario. Es utilizado en distribuciones Linux. Actualmente está siendo reemplazado por EXT4.

IP (Internet Protocol). Es un protocolo no orientado a conexión que permite el desarrollo y transporte de datagramas desde el origen al destino.

Log. Es un registro oficial de eventos durante un rango de tiempo en particular, es usado para registrar datos o información.

Linux. Desarrollado por Linus Torvalds, es un sistema operativo que puede ser distribuido, copiado y modificado gratuitamente.

LVS (Linux Virtual Server). Es un servidor de alta escalabilidad y disponibilidad construido en un clúster de servidores reales, con el equilibrador de carga que se ejecutan en el sistema operativo Linux[33].

Md5. Es un algoritmo criptográfico que para cualquier entrada produce una salida compleja de 16 bytes.

Nodo. Representa al número de computadoras instaladas en una red.

PILA LAMP. Acrónimo utilizado para referirse a un conjunto de subsistemas de software necesarios para alcanzar una solución global.

RPM. Sistema de administración y configuración de paquetes de software en plataformas Linux.

SHA1 (Secure Hash Algorithm). Es un algoritmo que realiza funciones hash criptográficas unidireccional.

[33] What is the Linux Virtual Server (20011) recuperado el 09 de Julio 2011 del sitio web http://www.linuxvirtualserver.org/

UDP (User Datagram Protocol). Es un protocolo de la capa de transporte del modelo TCP/IP. No orientado a conexión y permite el envío de datagramas a través de la red.

Unicast. Proceso de envió desde un único emisor a un único receptor.

URL (Uniform Resourse Locator). Dirección única que identifica a una página web en internet.

SSH. Es un programa que permite acceder a maquinas remotas a través de una red.