ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

INFORME DE MATERIA DE GRADUACIÓN

"Implementación de un sistema de VoIP de alta disponibilidad basado en Asterisk y Heartbeat"

Previa a la obtención del Título de:

LICENCIADO EN REDES Y SISTEMAS OPERATIVOS

Presentada por:

MAPY ASUNCION CASTILLO PALMA

ANA SOFIA ROCHA PEREIRA

GUAYAQUIL – ECUADOR

AÑO

2011

D E C L A R A C I Ó N E X P R E S A

“La responsabilidad del contenido de esta Trabajo de Grado, nos corresponde exclusivamente; y el patrimonio intelectual de la misma a la Escuela Superior Politécnica del Litoral”.

(Reglamento de Graduación de la ESPOL)

_____

Mapy Asunción Castillo Palma

_____

Ana Sofia Rocha Pereira

T R I B U N A L D E S U S T E N T A C I O N

_____

Ing. Gabriel Astudillo

PROFESOR DE LA MATERIA DE GRADUACIÓN

_____

Ing. Albert Espinal

PROFESOR DELEGADO POR EL DECANO DE LA FACULTAD

R E S U M E N

En los últimos años, la Telefonía IP se ha convertido en una de las principales tecnologías donde se implementan diferentes tipos de sistemas y servicios que sirven de soporte para múltiples actividades, las cuales generan dinero y se han vuelto esénciales para cualquier empresa.

Gracias a su creciente utilización, para la telefonía IP es indispensable contar con sistemas tolerantes a fallos, para que el servicio este siempre disponible, así como también nos permita aumentar la capacidad de procesamiento dependiendo a la demanda por el servicio.

Es por esta razón que el proyecto se enfoca en la Implementación de un sistema de VoIP basado en DRBD, Asterisk y Heartbeat capaz de ofrecer un servicio High-Availability.

INDICE DE CONTENIDO

INDICE DE FIGURAS

INDICE DE TABLAS

I N T R O D U C C I ON

Este proyecto fue realizado con la finalidad demostrar las distintas fases que se han seguido para la implementación de un sistema de telefonía basado en DRBD, Asterisk y Heartbeat

El servicio de telefonía implantado está dotado de alta disponibilidad en diversos aspectos. Uno de ellos está relacionado con los fallos tipo hardware como puede ser el fallo de la fuente de alimentación de uno de los nodos del sistema, corte del suministro eléctrico, etc. Ante estas situaciones el servicio de telefonía no se verá afectado.

Por otro lado, ante fallos tipo *software* (gracias a la integración con los servicios de alta disponibilidad) el servicio *VoIP* continuará funcionando como si nada hubiera ocurrido.

La central telefónica Open Source Asterisk, mediante DRBD y Heartbeat, proporciona un método eficaz para combatir los problemas de costos de implementación de un proyecto de esta magnitud, ya que permite realizarlo con equipos básicos y con software gratis.

CAPITULO 1

ANTECEDENTES Y JUSTIFICACION

1.1 ANTECEDENTES

Algunos de los clientes más exigentes que requieren comunicación a tiempo completo, como un Centro de Llamadas o la industria de la Banca, necesitan contar con una estructura física redundante que les provea el menor tiempo de caída, estos usuarios están muy lejos de aceptar una solución que no tenga mecanismos a prueba de fallos o alta disponibilidad. Para esto se necesita una inversión muy alta que muchas veces no está al alcance del presupuesto de una pyme que empieza en el mundo de los negocios.

Eso en la parte física de nuestra red en cuestión, por otro lado, en la estructura lógica de la red, se pueden implementar diversos tipos de soluciones combinando el hardware y software adecuado para satisfacer estas necesidades.

En el caso de una implementación de telefonía IP, usando equipos Linux y con Asterisk como “administrador” de nuestra centralita IP, las soluciones son varias entre ellas: DRBD, Heartbeat, LVS, etc. Pero estos proyectos proveen redundancia en puntos específicos del sistema.

Este proyecto se concentrara en dar “alta disponibilidad” a un sistema de VoIP basado en Asterisk usando Heartbeat.

Heartbeat es un mecanismo que provee alta disponibilidad a través de “clúster”, este término no está muy bien definido y puede tener diferentes significados para las personas; según el diccionario informático, un clúster es la unidad de almacenamiento en el disco rígido, muchos de los usuarios Windows estamos relacionados con la perdida de clusters que puede ser solucionado mediante la ejecución de la utilidad de desfragmentación.

Sin embargo, en un nivel más avanzado de nuestra industria, un clúster puede significar un grupo de computadoras conectadas entre sí de modo que se obtiene más “poder”, como por ejemplo se puede lograr ejecutar mas instrucciones por segundo, o se puede lograr una mayor disponibilidad.

1.2 JUSTIFICACIÓN:

Las soluciones informáticas de alta disponibilidad enfocadas a las grandes empresas son especialmente utilizadas en servicios críticos. Normalmente, estos servicios irán acompañados de complementos que requieren de un funcionamiento redundante para su eficaz proceso; por esta razón en vista al incremento de recursos económicos necesarios para implementar una central telefónica de un ambiente empresarial en un esquema privativo, vemos las

necesidades especificas del cliente. En donde la alternativa es un software libre para las implementaciones de redes de voz usando el estándar de comunicación VoIP.

No obstante a nivel empresarial se requiere tener esquematizado la integridad alta disponibilidad y alto rendimiento en un sistema VoIP de manera que permita adaptar las características del sistema a los requisitos que se deseen cumplir y manteniendo los costes de implantación.

1.3 DESCRIPCIÓN DEL PROYECTO

La implemetacion de nuestro proyecto puede alcanzar los siguientes objetivos:

1.3.1 OBJETIVO GENERAL

- Implementar en Linux un sistema de VoIP basado en clúster de alta disponibilidad, que le permitirá al usuario final tener redundancia en los servicios así como también brindar un servicio transparente.

1.3.2 OBJETIVOS ESPECÍFICOS

- Integración de DRBD Distributed Replicated Block Device con Asterisk y Heartbeat con el fin de proporcionar redundancia en varios puntos del sistema.
- Implementar un sistema tolerante a fallos, para asegurar la alta disponibilidad.
- Proveer comunicación rápida y precisa entre el Servidor Asterisk y los dispositivos de intercomunicación.
- Utilizar el modelo Heartbeat con alta disponibilidad a diseñar, para brindar escalabilidad a futuro y así aumentar la disponibilidad

1.4 METODOLOGÍA

Para cumplir con nuestros objetivos realizaremos la instalación de Asterisk sobre servidor real y un servidor virtual con sistema operativo CENTOS LINUX.

Adicionalmente dotaremos a los servidores con Heartbeat y Distributed Replicated Block Device para poder establecer canales de comunicación con alta disponibilidad y transparencia para el usuario.

1.5 PERFIL DE LA TESIS

Para el desarrollo de nuestro proyecto la plataforma que usaremos será Asterisk instalado en varios servidores con Centos sobre una estructura que usa teléfonos IP y softphones donde proveerá alta disponibilidad en la posible caída de uno de estos servidores.

En el capítulo 2 mencionaremos un análisis teórico de las funciones, características, mecanismos de implementación de cada una de las herramientas a utilizar para el desarrollo y mejor comprensión de nuestro proyecto.

En el capítulo 3 se indicara los detalles técnicos a implementarse en el análisis, diseño y la solución del proyecto.

Finalmente en el capítulo 4 se procederá a realizar las diferentes pruebas para validar el efectivo funcionamiento del servidor de respaldo; el cual nos garantiza que el servicio no sea interrumpido.

CAPITULO 2

DRBD HEARTBEAT Y ASTERISK

2.1 ASTERISK

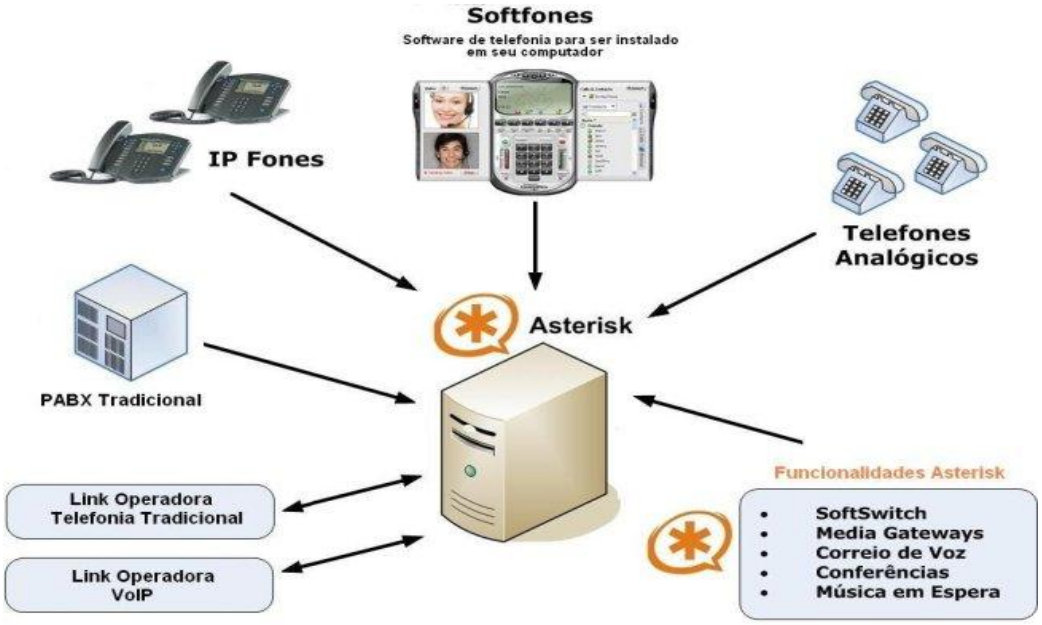

Fig. 1 Central Asterisk

Asterisk es una aplicación de software libre (bajo licencia GPL) de una central telefónica PBX (Private Branch Exchange y *Private Automatic Branch Exchange* para PABX), este software es una herramienta que nos ayuda a conectar directamente cualquier central telefónica a la red pública de teléfonos por medio de las líneas troncales actúa en Linux, BSD, Windows (emulado) y OS X proporcionando todas las funciones y características que se desarrollan en una PBX; también hace posible la utilización de VoIP en tres protocolos y puede interoperar con equipos de telefonía estándar usando un hardware relativamente sin costo. Facilita servicios de voicemail con directorios, conferencias, respuesta de voz interactivo IVR, llamadas en

espera. Tiene el soporte de tres tipos de formas de llamadas (tripartita): servicios de llamada con identificación, ADSI, SIP y H323 (como cliente y gateway), MGCP (administrador de llamadas solamente)

Para su total desempeño en Voz sobre IP no necesita de ningún hardware adicional pero para interconectarse con la telefonía tradicional se solicita el uso de las tarjetas especiales que se instalan en el computador con un costo muy económico llamadas FXO y FXS de 1 a 4 puertos muy comunes para instalaciones pequeñas.

Apoyando también una amplia gama de protocolos TMD para el manejo y transmisión de interfaces de la telefonía tradicional con tipo de señalización estándar americano y europeo en asuntos de sistemas de telefonía, permitiendo ser un nexo entre las redes integradas de datos de voz de siguiente generación y la infraestructura existente.

2.1.1 Funciones Básicas de Asterisk

Puede funcionar como cualquier centralita tradicional e incorpora todas sus funcionalidades. Enumeramos las más importantes:

- Conexión con líneas de telefonía tradicional, mediante interfaces tipo analógico (FXO) para líneas de teléfono fijo o bien móvil y RDSI (BRI o PRI).

- Soporte de extensiones analógicas, bien para terminales telefónicos analógicos, terminales DECT o bien equipos de fax.
- Soporte de líneas (trunks) IP: SIP, H323 o IAX.
- Soporte de extensiones IP: SIP, SCCP, MGCP, H323 o IAX.
- Música en Espera basada en archivos MP3 y similar.
- Funciones básicas de usuario:
    - Transferencias (directa o consultiva)
    - Desvíos
    - Capturas (de grupo o de extensión)
    - Conferencia múltiple
    - Aparcamiento de llamadas (Call parking)
    - Llamada directa a extensión
    - Retrollamada – Call back (llamada automática cuando disponible)
    - Paging - Megafonía a través del altavoz del teléfono[2]
    - DND

2.1.2 Escalabilidad de Asterisk.

- TDMoE (DIVISIÓN DE TIEMPO MÚLTIPLE SOBRE INTERNET): Permite la conexión directa con el PBX del Asterisk estado latente cero utiliza un hardware de Internet material
- VoIP: Permite la integración de instalaciones físicamente separadas utiliza conexiones de dato comúnmente desplegados permite un plan unificado a través de múltiples oficinas.

2.2 CLUSTER

2.2.1 Clústeres en general:

El termino clúster es a menudo usado para diferentes propósitos y existen diferentes definiciones para este. En química, un clúster es una conformación temporal de átomos que otorga características particulares a compuestos metálicos. Es más, en Física, un clúster es un grupo relacionado de estrellas que se mantienen juntas por esta interacción. Existen muchos otros usos para la palabra clúster en varias ciencias.

En la ciencia informática, esto es incluso peor. Existe un vasto número de definiciones y especialmente opiniones de cómo definir un clúster.

En resumen, llamamos clúster a un conjunto de equipos conectados entre sí que se encuentran brindando uno o más servicios en común. De esta forma se logra ofrecer redundancia a nivel del servicio, eliminando la existencia de un único punto de falla.

Esta configuración es transparente para los usuarios, ya que desde su punto de vista solo existe un equipo al cual se están conectando.

Otro de los beneficios de esta tecnología es poder dar un mejor uso a los recursos de los equipos que forman parte del clúster. Se logra balancear la carga que es recibida, de forma que ninguno de los miembros este siendo subutilizado cuando otro se encuentra haciendo un uso excesivo de los recursos.

Debido a esto la distinción entre sistemas distribuidos y un clúster es más difícil que la de un sistema en paralelo. En general, un sistema distribuido está compuesto de varios “tiers” o niveles. Un tier es por esto un poco de la capa del sistema distribuido con una responsabilidad específica.

Un famoso ejemplo para un sistema distribuido de dos niveles es la arquitectura cliente-servidor por el cual el servidor ofrece un servicio al cliente (ejemplo: un servidor web). En el campo de los sitios web, el navegador web del cliente representa el primer nivel, y el segundo nivel lo representa el servidor web. Si el servidor web ofrece

contenidos entrantes de un servidor de bases de datos separado, entonces el servidor de bases de datos es el tercer nivel.

Teniendo esto en cuenta, el tiempo de caída de un servidor de base de datos por un día puede costarle a una compañía miles de dólares e incluso más. Por lo tanto, los sistemas basados en clusters de Alta Disponibilidad son establecidos para garantizar una cantidad segura de disponibilidad por redundancia.

2.2.2 Categorías / Tipos de Clusters:

Existen 3 áreas mayormente usadas para clusters:

- High-Performance Computing (HPC) Clusters
- Load-Balancing (LB) clusters
- High-Availability (HA) clusters

2.2.2.1 High-Performance Computing (HPC) Clusters

Estos clusters son muy populares y probablemente una de las primeras imágenes que se nos viene a la mente cuando escuchamos hablar del término clúster: un amplio conjunto de computadoras en un sótano oscuro teniendo que calcular miles de peticiones por segundo. En realidad, esta es una de las formas

más antiguas de implementar un clúster en centros de computación.

La mayoría de las computadoras en el mundo están construidas con el concepto de “procesamiento en paralelo de alta velocidad”, en poderosas computadoras esto es alcanzado mediante la unión de la potencia de cada computadora individual. Un ejemplo de clúster lo podemos encontrar en “Deep Blue” la súper computadora hecha por IBM, que jugó ajedrez con el campeón del mundo Garry Kasprov, este fue un clúster que consistía de varios cientos de RS6000s. De hecho, muchas de las grandes compañías de animación de Hollywood, como Pixar, Industrial Light y Magic, usan clusters de computadoras considerablemente por la interpretación (el proceso de traducir toda la información como color, movimiento, propiedades físicas, etc., en un simple cuadro de imagen).

En el pasado, una supercomputadora era un artículo de lujo y por sobre todo costoso que solo algunas pocas universidades o centros de investigación podían costear.

2.2.2.2 Load-Balancing (LB) clusters

Es una técnica que nos permite distribuir la carga de CPU, red, disco, memoria u otro componente, de forma proporcionada a través de un conjunto de equipos interconectados.

De esta forma se hace un mejor uso de los recursos y se minimiza la sobre carga que pueda existir debido a una alta demanda por parte de los usuarios de los servicios. Esta configuración brinda también un nivel de redundancia ya que al contar con más de un equipo que se encuentra ofreciendo el mismo servicio, es posible sobrellevar la caída de uno de ellos sin que el servicio se vea afectado.

Es posible balancear carga haciendo uso de un hardware especializado o un software que brinde dicha funcionalidad. En cualquiera de los casos el funcionamiento es casi el mismo, con la diferencia que en el caso de un software, luego de definir el conjunto de equipos que tienen el servicio que va a ser balanceado, se configura un nombre DNS y se lo asocia a una dirección de IP que va a representar un equipo virtual al cual los usuarios van a ser direccionados; en el caso de un Appliance el nombre y la IP son configurados en él.

Algunos de los servicios comúnmente balanceados son: páginas Web, CRMs, correos, aplicaciones de terceros que requieren acceso web, entre otros.

En ocasiones también se combinan las funcionalidades de balanceo de carga con las de un clúster de servicios para brindar además de una distribución de los recursos, redundancia en el acceso a la base de datos del servicio.

2.2.2.3 High-Availability (HA) clusters

Las soluciones de alta disponibilidad enfocada a los servicios son especialmente utilizadas en servicios críticos. Normalmente, estos servicios irán acompañados de complementos que se requieren para su funcionamiento tales como: direccionamiento IP, reglas de firewall, copias de seguridad, etc. Por ejemplo, la mayoría de las empresas requieren de la utilización del servicio de Directorio Activo por lo que este se convierte en este caso en un servicio crítico. Una configuración de alta disponibilidad podría consistir por ejemplo en dos nodos donde solo uno de ellos dispone de una dirección IP Virtual, mediante la cual las aplicaciones acceden al Directorio Activo. En caso de fallo del nodo que ofrece el servicio, la dirección IP Virtual se establece en el otro nodo (nodo secundario o backup) lo que permite a los clientes y aplicaciones

seguir accediendo al Directorio Activo como si no hubiera ocurrido nada.

Para nuestro estudio un clúster en la categoría de Alta Disponibilidad, usan varias tecnologías para ganar un nivel extra de fiabilidad para un servicio. Compañías como Red Hat, Turbolinux y PolyServe tienen productos de clúster que podrían permitir a un grupo de computadoras monitorearse entre si, cuando un servidor maestro (como un web server) cae, es decir deja de funcionar, un servidor secundario tomará el control de los servicios similar al “disk mirroring” entre servidores. Un clúster podría servir también en el caso de un problema de hardware, nuestros usuarios tendrían igualmente servicio ya que uno de los nodos tomaría la posta como maquina primaria.

Para esto, usaremos Heartbeat que es un paquete de software creado por LINUX-HA, funciona de forma similar al System V o init pero en vez de una sola maquina pasaría a ejecutar los servicios en los nodos, basándose en que no llegan respuestas estas se hacen por medio de ping y por pulsaciones del cable serie.

2.3 DISTRIBUTED REPLICATED BLOCK DEVICE

2.3.1 Definición:

Es un dispositivo de bloque diseñado para construir clústeres de alta disponibilidad. Esto se hace replicando todo un dispositivo de bloque sobre una red. Se podría ver como una implementación de raid-1 sobre una red. También conocido como un sistema de almacenamiento replicado que utiliza la semántica de un sistema de almacenamiento compartido:

- Dispositivo de Bloque
- Replicación
- Distribuido

2.3.1.1 Dispositivo de Bloque, se refiere al lugar donde podemos colocar un sistema de archivos (como ext3). En los sistemas Linux, un dispositivo de bloque estaría referenciado como ***/dev/sda***.

2.3.1.2 Replicación, se refiere a que cualquier cambio en el dispositivo de bloque local se copia a los nodos adicionales en tiempo real para incrementar la disponibilidad de los datos.

2.3.1.3 Distribuido, se refiere a que se reduce el riesgo de la pérdida de datos, propagando los datos a nodos en diferentes lugares.

El propósito de DRBD es mantener los datos disponibles, aun cuando un servidor de datos falle completamente, utilizando un clúster de alta disponibilidad.

2.3.2 Elementos para construir un Clúster de Alta Disponibilidad con DRBD

Para un modelo de DRBD, se necesitan dos nodos que implementen DRBD, Heartbeat y una aplicación que trabaje sobre el dispositivo de bloque, es decir, un sistema de archivos.

Aquí DRDB toma el control de los datos, los escribe en el disco local del primer nodo y los manda al segundo nodo. En el otro nodo se guardan los datos en el disco.

Usualmente DRBD es controlado por Heartbeat. Aquí Heartbeat inicia el proceso a prueba de fallos, es decir, que cuando el nodo principal

falla el segundo nodo tomará el control, siendo totalmente transparente para el usuario.

2.3.3 Funcionamiento de DRBD

Cada uno de los nodos implementará un dispositivo de bloque y estos tendrán un estado, que puede ser primario o secundario. Cuando una aplicación quiera acceder a los datos, solo podrá hacerlo al nodo que tenga el dispositivo de bloque primario (***/dev/drbdX***). DRBD permite implementar más de un solo dispositivo de bloque, y cada uno puede servir para diferentes aplicaciones.

Cuando la aplicación haga una escritura en el nodo primario, los datos serán enviados al dispositivo de bloque y luego serán enviados al dispositivo de bloque del nodo con el estado secundario por medio de la red. Es este nodo secundario, simplemente se escribirán los datos en el dispositivo de bloque.

En caso que el nodo principal falle, *Heartbeat* cambiará el estado del dispositivo de bloque secundario (que se encuentra en el nodo secundario) a un estado primario y este segundo nodo tomará el control. Ahora, todas las aplicaciones accederán a los datos que se encuentran en este segundo nodo.

En caso que el primer nodo regrese a funcionar, este se convertirá en el nuevo nodo secundario y tendrá que sincronizar su contenido con el nodo primario. La sincronización que se efectúa aquí, pero solo sincronizará las partes del dispositivo de bloque que hayan sido modificadas. DRBD siempre hará una re-sincronización inteligente, cada vez que sea posible. Aquí se puede definir una configuración activa de cierto tamaño que hará posible un tiempo de re-sincronización total de 1 a 3 minutos sin importar el tamaño del dispositivo de bloque.

2.4 HEARTBEAT

2.4.1 Definición

Es un proyecto Open Source fundado por Alan Robertson en 1999. El objetivo de este proyecto es proveer software de clustering de alta disponibilidad (HA) para Linux y otras plataformas. El proyecto implementa el estándar Open Cluster Framework (OCF). Es un software que ofrece alta disponibilidad a determinados recursos mediante la creación y mantenimiento de un clúster compuesto por una serie de nodos. Los recursos se ejecutan y se mueven entre los distintos nodos ya sea por motivos de fallo o simplemente por motivos de administración. Por ejemplo un recurso podría ser el servicio Web

de una entidad de tal manera que si sucede un evento como el fallo de la fuente de alimentación del nodo que en ese momento ejecuta el servicio, el servicio Web o recurso se mueve hacia otro nodo del clúster por lo que el servicio en cuestión se continua ofreciendo de manera ininterrumpida a pesar del suceso.

Linux-HA es un protocolo que provee un framework flexible, de Alta Disponibilidad. Este protocolo desempeña la función de enviar mensajes en intervalos regulares de tiempo entre máquinas y si un mensaje no es recibido de una máquina en particular, entonces se asume que la máquina ha fallado y toma alguna forma de acción evasiva.

Las plataformas que soportan Heartbeat son las distribuciones de Linux como: SuSe, Mandriva, Debian, Ubuntu, Red Hat y Gentoo. Otras plataformas incluyen: FreeBSD, OpenBSD, Sun Solaris y Mac OS X.

La segunda versión de Heartbeat ha sido modificada para eliminar ciertas restricciones presentes en la primera versión, las cuales presentaban grandes limitaciones en su funcionamiento. Las restricciones más importantes que limitaban la funcionalidad de la primera versión de Heartbeat son las siguientes:

- Clúster con un tamaño máximo de dos nodos.

- Incapacidad de monitorizar los recursos. Heartbeat-1 no monitoriza los recursos con la finalidad de comprobar si los recursos están operando correctamente por lo que solo se supervisa el estado de funcionamiento del nodo (monitorización hardware) sin tener en cuenta el estado de ejecución de los recursos (monitorización software). De esta manera si el nodo que ofrece el servicio contesta adecuadamente al ping indicando “estoy vivo” pero el recurso, por ejemplo un servicio Web, no se está ejecutando correctamente o incluso ha parado, no se toma ninguna medida de recuperación. Para monitorizar recursos con Heartbeat-1 es necesario hacer uso de aplicaciones de monitorización externas tales como Watchdog.

- Mínima capacidad de expresar la información dependiente. En Heartbeat-1 no es posible crear grupos de recursos que compartan las mismas restricciones, grupos de nodos, etc., lo que repercuten en una pobre flexibilidad a la hora de configurar el clúster.

Para poder construir un modelo activo / pasivo, al menos se necesitan dos máquinas que ejecuten Heartbeat simultáneamente. La primera de las máquinas, que se le conoce como maestro, es la que ofrece el servicio normalmente. La segunda máquina, conocida como esclava,

es la que suplantará a la máquina principal o maestro en caso que esta deje de funcionar o de prestar sus servicios. Heartbeat nos da la posibilidad de soportar n nodos en este modelo de activo / pasivo.

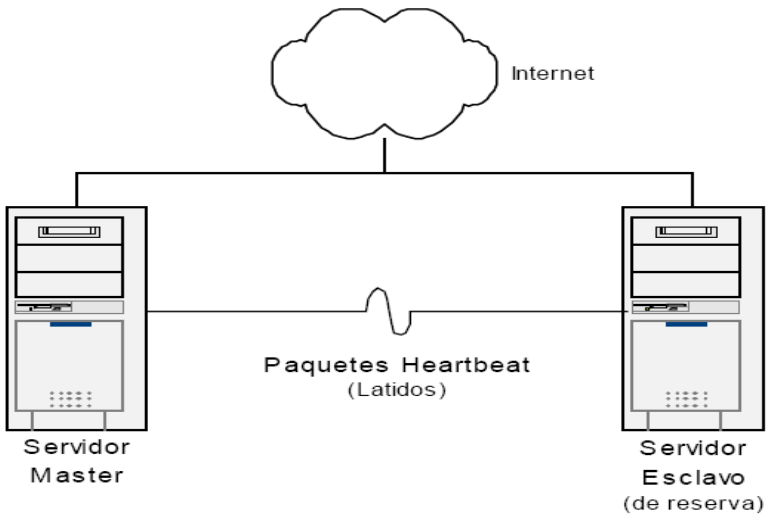

Fig. 2 Diseño de Heartbeat

2.4.2 Funcionamiento:

El funcionamiento de Heartbeat se basa en el envío y recepción de señales enviadas por los demonios de Hearbeat que se ejecutan en ambas máquinas, a estas señales se les conocen como “latidos”.

La diferencia entre el servidor maestro y el esclavo, radica en la prioridad que tiene para ofrecer un servicio. Aquí, el esclavo solo prestara el servicio cuando deje de escuchar los latidos del maestro durante un periodo de tiempo determinado, pasado el cual se supondrá que servidor maestro dejó de funcionar.

Una vez que el esclavo vuelva a escuchar los latidos del maestro, este tomará el control nuevamente, a menos que dentro de la configuración de Heartbeat se haya colocado la directiva auto_failback en off. Esta directiva puesta en off, quiere decir que si la máquina que era maestro vuelve a funcionar, ya no retomará el control del servicio, sino se convertirá en la nueva esclava.

El maestro y esclavo pueden comunicarse por medio de dos interfaces, el puerto serial (RS-232) o las tarjetas Ethernet.

Puede darse el caso de un error en la interfaz que une a ambas máquinas que imposibilita la llegada de latidos hacia el esclavo. Si sucede esto, el esclavo interpretará erróneamente que el servidor maestro ha caído y tratara de tomar el control de la prestación del servicio, cuando el servicio nunca ha dejado de prestarse. Para solucionar este error o simplemente para reducir el riesgo de que suceda el fallo de las interfaces es posible utilizar la técnica de Bonding entre las interfaces.

Heartbeat implementa una serie de tipos de latidos:

- Bidirectional Serial Rings.
- UDP/IP Broadcast, UDP/IP Multicast.
- Pings especiales Heartbeat para routers.

Heartbeat es el encargado de inicializar o detener el servicio al cual se le quiere prestar en alta disponibilidad. Este servicio es prestado a

través del uso de una única IP en ambas máquinas, que en realidad no es mas que la dirección IP Virtual (VIP). A esta VIP se le considera como un recurso. Los recursos son encapsulados como programas que trabajan de una manera similar a los scripts de inicialización de Linux, esto significa que los recursos puede se inicializados o terminados, o que se les puede preguntar si están siendo ejecutados o no.

Aquí ambas máquinas, tanto la esclava como la maestra, tienen una dirección IP propia y también comparten la VIP. Cuando Heartbeat lanza el servicio a prestarse, se activa la VIP en esa máquina.

El proceso mediante el cual una máquina se adueña de la VIP que estuvo utilizando la otra, se le llama TakeOver. Este proceso solo puede considerarse exitoso cuando la máquina que deja el servicio desactiva la VIP de su interfaz, y la máquina que toma el servicio, en vez de la otra, activa la VIP en su interfaz. Para esto, las otras máquinas necesitan actualizar sus tablas ARP, por lo cual Heartbeat envía peticiones y respuestas ARP innecesarias, que se les conoce como ARP gratuitos, para forzar que las otras máquinas actualicen su tabla ARP con la nueva dirección MAC de la VIP.

Entonces, cuando se produce un fallo en la máquina que presta el servicio se le conoce como failover. Cuando el maestro vuelve a estar activo se le conoce como failback. Existe un demonio, llamado

ldirectord, que sirve para monitorear servidores reales así como para administrar las tablas de ruta de los servidores que implementan Heartbeat. A este demonio también se le considera un recurso.

CAPITULO 3

IMPLEMENTACION

3.1 INTRODUCCION

El fin de proyecto de graduación es conocer y aplicar los beneficios del software libre, ya que nos permite disminuir costos y alta disponibilidad al implementar soluciones en el mundo real.

Esta solución está basada en software libre asterisk que permite implementar centrales telefónicas a pequeña, mediana y gran escala. Con el uso de los diferentes agregados se puede ampliar la utilidad de dicho software y encontrar recursos para solucionar toda clase de problemas como lo resolverían las centrales telefónicas privadas.

La central telefónica tradicional es reemplazada por un computador que puede variar según las necesidades del cliente; el tamaño de la central dependerá de la concurrencia de llamadas que vaya a tener, pero siempre el dinero gastado será inferior que si compráramos una central telefónica.

3.2 HARDWARE

Características técnicas mínimas recomendadas para operar Centos:

- **Memoria RAM**: 192 MB (Mínimo).
- **Espacio en Disco Duro** : 850 MB (Mínimo) - 2 GB (Recomendado)

- **Procesador**: Intel Pentium I/II/III/IV/Celeron, AMD K6/II/III, AMD Duron, AMD Athlon/XP/MP ya que funciona sobre las redes IP, que son más baratas que las redes de almacenamiento especiales.

3.2.1 Servidor

Los requisitos de hardware que se han configurado para el desarrollo del proyecto, son los siguientes:

Tabla I Características de Servidores

Servidor#1		Servidor#2	
Procesador	Dual-Core	Procesador	Core 2 Duo
RAM	1 GB	RAM	1 GB
Disco Duro	15 GB	Disco Duro	15 GB
Tarjeta de Red	10/100 Mbps	Tarjeta de Red	10/100 Mbps

3.2.2 Switch

La conexión de los servidores se realizara por medio de un Switch Linksys de las siguientes características:

Tabla II Características de Switch D-Link

Velocidad de transferencia de datos	100 Mbps
Protocolo de interconexión de datos	Ethernet, Fast Ethernet
Interfaces	8 x red - Ethernet 10Base-T/100Base-TX - RJ-45 hembra – 8

3.3 SOFTWARE

3.3.1 Servidor PBX

El servidor que será utilizado como centralita telefónica tendrá los siguientes componentes instalados:

Tabla III Servidor PBX

Sistema Operativo	Centos Linux 5.4
Software IP PBX	Asterisk Verison 1.6.0.15-1
Dahdi Linux	DAHDI LINUX 2.4.0

Librerías necesarias para que Asterisk funcione correctamente como Clúster High Availability:

- drbd
- kmod-drbd82

- OpenIPMI-libs
- heartbeat-pils
- openhpi
- heartbeat
- heartbeat-stonith

3.3.2 Softphones

Los softphones son simuladores de teléfonos inteligentes, es decir nos permite usar la computadora para hacer llamadas a otros softphones o a otros teléfonos convencionales usando VoIP.

En este proyecto se ha utilizado los Softphones X-Lite y Zoiper simuladores de extensiones SIP.

3.4 INSTALACION

3.4.1 Instalación de Centos

Inicializar su sistema con el Disco de Centos 5.4 colocado en su lector de CD-ROM. Luego desde el menú de inicio, seleccionar “opciónes

avanzadas”; donde elegirá particionar manualmente el disco duro, el nuestro está basado en 15 GB.

- Crear una partición llamada root (/), con formato ext3 y 5120 MB (sda 1)
- Crear una partición swap con 5120 MB (sda 2)

Continuar con el resto de la rutina de instalación estándar de Centos.

Una vez terminada la instalación ejecutamos en la terminal el comando #yum –y update con la finalidad de actualizar todos los paquetes.

Este procedimiento debemos realizarlo exactamente igual en ambos servidores con el propósito de no tener errores posteriormente.

3.4.2 Instalación de DRBD – ASTERISK - HEARTBEAT

En la ruta /boot/grum/menú debemos asegurarnos que en el parámetro default sea igual a 1 para poder inicializar con non-xex kernel.

A continuación vamos a crear una partición, en la que se encontraran los datos a replicarse. En la terminal ejecutamos el comando:

- #disk /dev/sda

Aquí nos aparecerá un mensaje para añadir una nueva partición donde seteamos la letra (n). Luego tendremos que elegir el tipo de partición a crear, por lo optaremos por la opción de primario digitamos la letra (p).

A continuación nos mostrara un mensaje para escoger el número de partición, en la cual nos decidiremos por (3).

Presionamos enter hasta que regresar a fdisk command prompt, y proseguir con la instalación

Si los servidores tienen dos diferentes tamaños de disco duro es imperativo que la tercera partición sea idéntica en tamaño o nunca se sincronizara DRBD. Se debe realizar este procedimiento aceptando el cilindro por defecto y luego especificar el ultimo cilindro con la opción +sizeM. Ex +6048M

Presione la letra “t” para cambiar el ID del sistema de partición.

Presione el numero “3” para escoger el numero de partición.

Elegimos “HEX 83” para el tipo de partición a instalarse.

Presionamos la letra “w” para guardar cambios.

Una vez realizado este procedimiento reinicie los servidores para actualizar todos los cambios realizados.

Este procedimiento se debe ejecutarlo con las mismas especificaciones en ambos servidores garantizar el correcto funcionamiento de la instalación.

Le proporcionamos formato a nuestra partición ejecutando el siguiente comando:

- #mke2fs –j /dev/sda3

Ahora debemos eliminar el sistema de archivos del disco que acabamos de crear con el comando:

- dd if=/dev/zero bs=1M count=1 of=/dev/sda3; sync

Instalamos DRBD, Heartbeat y sus dependencias con yum. Ejecutar los siguientes comandos:

- yum install drbd
- yum install kmod-drbd82

- yum install OpenIPMI-libs
- yum install heartbeat-pils
- yum install openhpi
- yum install Heartbeat
- yum install Heartbeat-stonith

Para garantizar el nombre de host adecuado para la resolución de IP, se recomienda actualizar manualmente el archivo /etc/hosts para reflejar la adecuada propiedad intelectual de asignación. Como muestra la figura 3.4

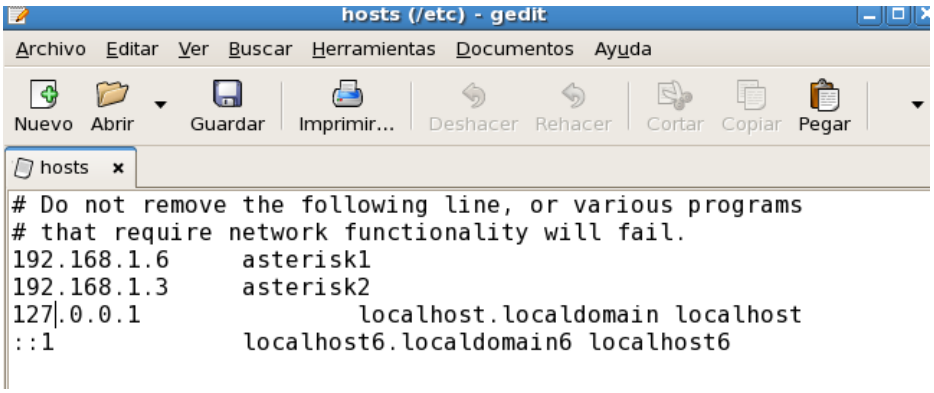

Fig. 3 Archivo /etc/hosts

Editamos el archivo /etc/drbd.conf, el cual debe ser idéntico en ambos servidores. DRBD utiliza tres protocolos: A, B y C, en nuestro caso hemos utilizando el Protocolo C. Este protocolo de comunicación antes de grabar los datos al disco local lo hace al disco en red garantizando la seguridad de los datos y demostrado ser muy rápido. Como muestra la figura 3.5

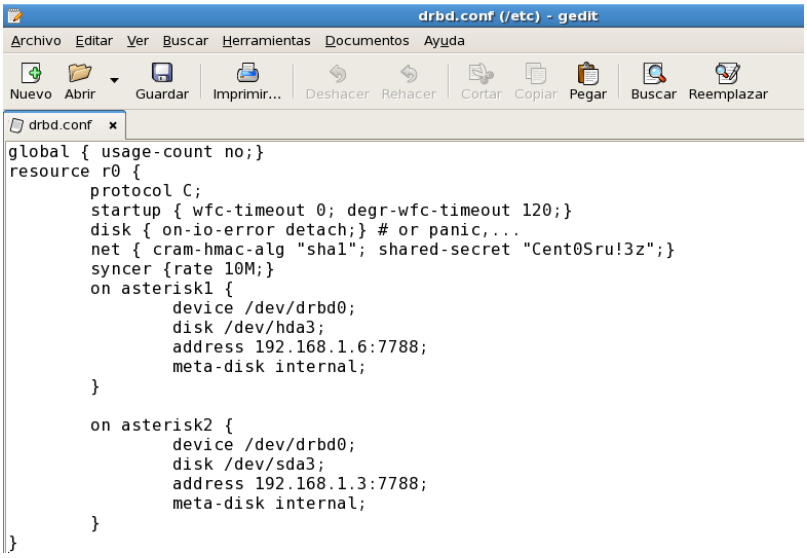

Fig. 4 Archivo /etc/drbd.conf

En ambos servidores creamos una partición virtual /dev/drbd0. Ejecutamos el comando:

- #drbdadm create-md r0

El proceso de sincronización se realiza al inicializar el servicio de drbd en ambos servidores:

- #service drbd start

Luego de unos minutos verificamos el proceso de sincronización con el siguiente comando:

- #service drbd status

Inicialmente ambos servidores son “secundario”; necesitamos asignar quien es el servidor “primario”. Ejecutamos el siguiente comando solo en servidor que vayamos a denominar como “primario”. En nuestro caso lo realizamos denominados a “asterisk1”:

- #drbdsetup /dev/drbd0 primary –o

Solo en el servidor “primario”, se monta la partición vitual /dev/drbd0, pero primero debemos formatear la partición con ext3 usando los siguientes comandos:

- #mke2fs –j /dev/drbd0
- #mkdir /replica
- #mount /dev/drbd0 /replica

Modificamos los archivos en el servidor “asterisk1” tal como muestran las figuras:

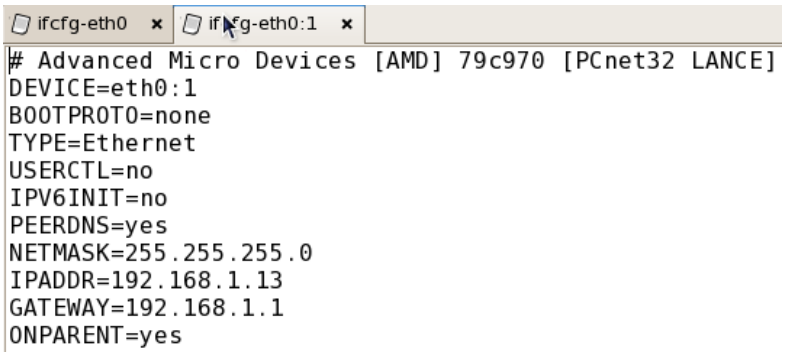

Fig. 5 Archivo de configuración ifcfg-eth0:1

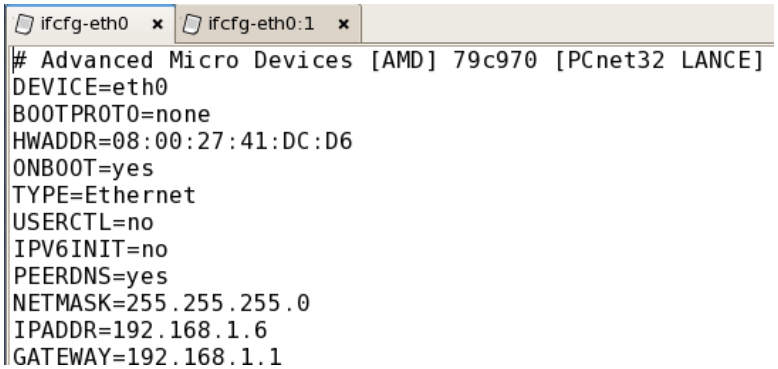

Fig. 6 Archivo de configuración ifcfg-eth0:0

Solo en el servidor “asterisk1”, se procede a copiar todos los directorios que deseamos se sincronicen entre los dos servidores a nuestra nueva partición. Removemos los directorios originales y creamos symbolic link para reemplazarlos.

- #/etc/asterisk
- #/var/lib/asterisk
- #/usr/lib/asterisk
- #var/spool/asterisk
- #var/log/asterisk

Para la instalación de asterisk realizamos el siguiente procedimiento en ambos servidores:

- Desde la pagina: http://www.asterisk.org/downloads/yum
- El primer paso es añadir yum repositories para su sistema Centos. Esto se hace mediante la creación de una entrada en el

directorio de configuración de yum (/etc/yum.repos.d por defecto). Utilice el editor de texto de su elección para crear un nuevo archivo llamado "centos-asterisk.repo" en el directorio "/ etc / yum.repos.d" carpeta. Agregar el siguiente texto:

```
[asterisk-tested]
name=CentOS-$releasever - Asterisk - Tested
baseurl=http://packages.asterisk.org/centos/$releasever/tested/$basearch/
enabled=0
gpgcheck=0
#gpgkey=http://packages.asterisk.org/RPM-GPG-KEY-Digium

[asterisk-current]
name=CentOS-$releasever - Asterisk - Current
baseurl=http://packages.asterisk.org/centos/$releasever/current/$basearch/
enabled=1
gpgcheck=0
#gpgkey=http://packages.asterisk.org/RPM-GPG-KEY-Digium
```

- Guardar el archivo y crear otro denominado: "centos-digium.repo" e insertar el texto siguiente:

```
[digium-tested]
name=CentOS-$releasever - Digium - Tested
baseurl=http://packages.digium.com/centos/$releasever/tested/$basearch/
enabled=0
gpgcheck=0
#gpgkey=http://packages.digium.com/RPM-GPG-KEY-Digium
```

```
[digium-current]
name=CentOS-$releasever - Digium - Current
baseurl=http://packages.digium.com/centos/$releasever/current/$basearch/
enabled=1
gpgcheck=0
#gpgkey=http://packages.digium.com/RPM-GPG-KEY-Digium
```

- En este punto el sistema se ha actualizado; para iniciar la instalación, ejecutamos el siguiente comando:

```
# yum install asterisk16 asterisk16-configs asterisk16-voicemail dahdi-linux
dahdi-tools libpri
```

Ahora vamos a copiar todos los directorios que se sincronizaran entre los dos servidores a nuestra nueva partición. Realizamos este procedimiento solo en el servidor “asterisk1”; eliminar los directorios originales y crear enlaces simbólicos para reemplazarlos en server1.drbd:

- *cd /replica*
- *tar -zcvf etc-asterisk.tgz /etc/asterisk*
- *tar -zxvf etc-asterisk.tgz*
- *tar -zcvf var-lib-asterisk.tgz /var/lib/asterisk*
- *tar -zxvf var-lib-asterisk.tgz*
- *tar -zcvf usr-lib-asterisk.tgz /usr/lib/asterisk/*
- *tar -zxvf usr-lib-asterisk.tgz*
- *tar -zcvf var-spool-asterisk.tgz /var/spool/asterisk/*

- *tar -zxvf var-spool-asterisk.tgz*
- *tar -zcvf var-log-asterisk.tgz /var/log/asterisk/*
- *tar -zxvf var-log-asterisk.tgz*
- *rm -rf /etc/asterisk*
- *rm -rf /var/lib/asterisk*
- *rm -rf /usr/lib/asterisk/*
- *rm -rf /var/spool/asterisk*
- *rm -rf /var/lib/mysql/*
- *rm -rf /var/log/asterisk/*
- *ln -s /replica/etc/asterisk/ /etc/asterisk*
- *ln -s /replica/var/lib/asterisk/ /var/lib/asterisk*
- *ln -s /replica/usr/lib/asterisk/ /usr/lib/asterisk*
- *ln -s /replica/var/spool/asterisk/ /var/spool/asterisk*
- *ln -s /replica/var/lib/mysql/ /var/lib/mysql*
- *ln -s /replica/var/log/asterisk/ /var/log/asterisk*

Recuerde detener cualquier arranque de los servicios en los servidores que deben ser controlados por Heartbeat. En nuestro caso será asterisk:

- #chkconfig asterisk off
- #service asterisk stop

Configuramos el archivo /etc/ha.d/ha.cf en el servidor “asterisk1”:

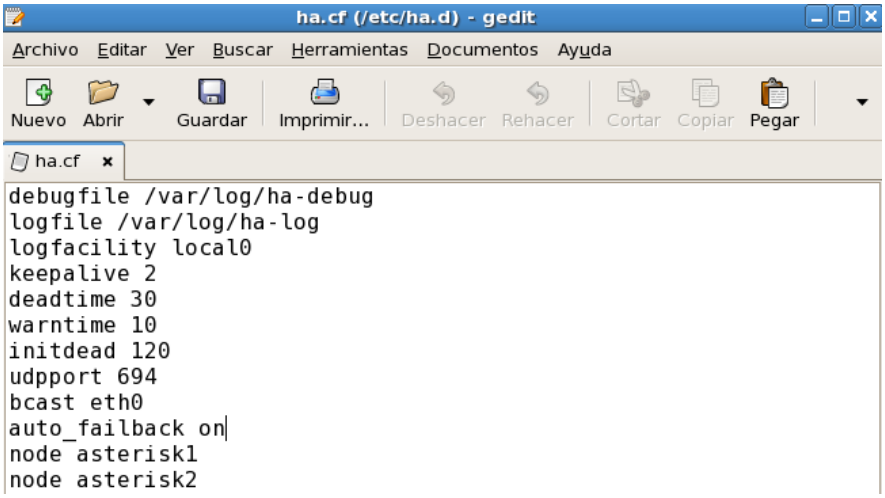

Fig. 7 Archivo /etc/ha.d/ha.cf

Configuramos el archivo /etc/ha.d/authkeys en el servidor “asterisk1”:

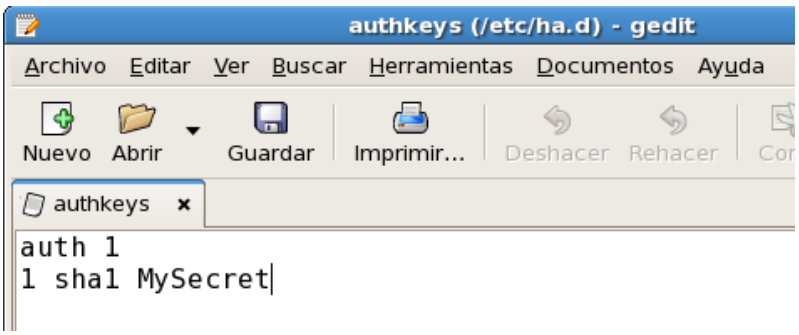

Fig. 8 Archivo /etc/ha.d/authkeys

Cambiamos los premisos en el archivo /etc/ha.d/authkeys en el servidor “asterisk1”:

- #chmod 600 /etc/ha.d/authkeys

Editamos el archivo /etc/ha.d/haresource en el servidor “asterisk”:

Fig. 9 Archivo /etc/ha.d/haresources

Inicializamos el servicio de Heartbeat en el servidor “asterisk1”:

- #service heartbeat start

Replicamos el archivo ha.cf, authekeys y haresources al servidor “asterisk 2” desde el servidor “asterisk1”:

- # scp /etc/ha.d/ha.cf /etc/ha.d/authkeys /etc/ha.d/haresources root@asterisk2.drbd:/etc/ha.d/

Inicializamos el servicio de Heartbeat en el servidor “asterisk2”:

- #service heartbeat start

Configurar Heartbeat para iniciar el arranque en ambos servidores:

- #chkconfig –add heartbeat

Reiniciamos el servicio drbd en ambos servidores:

- #sevice drbd restart

Verificar en el servidor “asterisk1” el estatus de drbd:

- #service drbd status

```
[root@asterisk1 ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.2.6 (api:88/proto:86-88)
GIT-hash: 3e69822d3bb4920a8c1bfdf7d647169eba7d2eb4 build by buildsvn@c5-i386-bui
ld, 2008-10-03 11:42:32
m:res  cs         st                ds                 p  mounted   fstype
0:r0   Connected  Primary/Secondary  UpToDate/UpToDate  C  /replica  ext3
```

Fig. 10 Estatus drbd en asterisk1

Verificar en el servidor “asterisk2” el estatus de drbd:

- #service drbd status

```
[root@asterisk2 ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.2.6 (api:88/proto:86-88)
GIT-hash: 3e69822d3bb4920a8c1bfdf7d647169eba7d2eb4 build by buildsvn@c5-i386-bui
ld, 2008-10-03 11:42:32
m:res  cs         st                ds                 p  mounted  fstype
0:r0   Connected  Secondary/Primary  UpToDate/UpToDate  C
```

Fig. 11 Estatus drbd en asterisk2

Verificar en ambos servidores el estatus de asterisk y Heartbeat:

- service asterisk status
- service heartbeat status

```
[root@asterisk1 ~]# service heartbeat status
heartbeat OK [pid 2062 et al] is running on asterisk1 [asterisk1]...
[root@asterisk1 ~]# service asterisk status
Se está ejecutando asterisk (pid 2536)...
```

Fig. 12 Estatus de Heartbeat y Asterisk en asterisk1

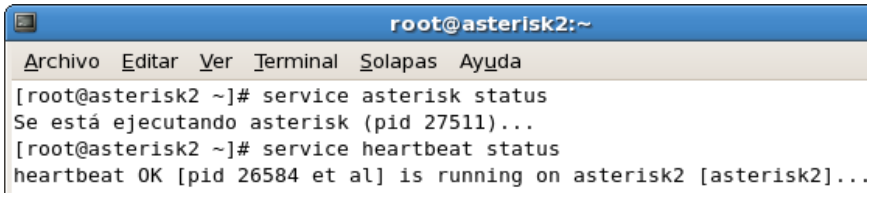

Fig. 13 Estatus de Heartbeat y Asterisk en asterisk2

3.4.3 Configuración de Archivos de Asterisk

3.4.3.1 Configuración SIP.CONF

En este archivo se configuran todas la extensiones que van a usar el protocolo SIP.

/etc/asterisk/sip.conf

[202]

Número de la extensión

type=friend

Tipo de extensión. Puede ser friend, user o peer. Friend puede hacer y recibir llamadas, user solo recibir y peer solo puede hacer (como en el caso de proveedores VoIP que usamos solo para hacer llamadas)

secret=202

Define la contraseña de la extensión

host=dynamic

Si la extensión se conecta remotamente cambiando continuamente su dirección IP se pone este parámetro

context=internal

El contexto que usara la extensión.

qualify=yes

Determina el tiempo de respuesta de una extensión y si está alcanzable o no

canreinvite=no

Si queremos que la extensión intente conectarse directamente con la extensión llamada: No; si queremos que Asterisk haga de puente entre las dos extensiones colocamos Yes

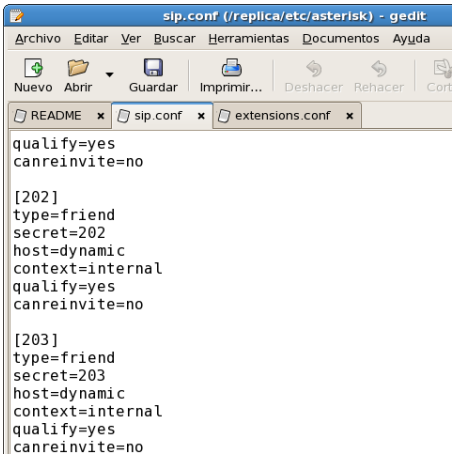

Fig. 14 Archivo sip.conf

3.4.3.2 Configuración extensiones

Este archivo contiene el plan de marcado de la central telefónica. El archivo extensions.conf es el más importante del Asterisk y tiene como misión principal definir el dialplan o plan de numeración que seguirá la centralita para cada contexto y por tanto para cada usuario.

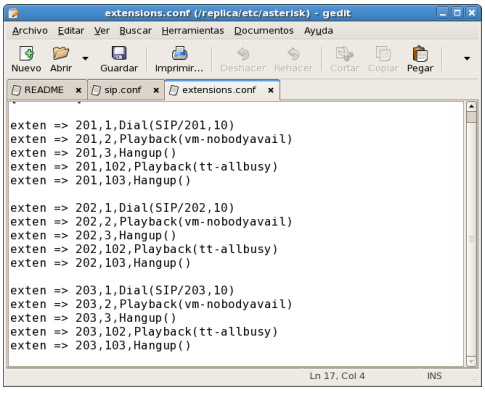

Fig. 15 Archivo extensions.conf

3.4.3.3 Configurando Softphone XTEN-XLITE

Para la configuración del /Xten-xlite primero procedemos a ejecutarlo ubicándonos en la carpeta donde lo hemos descomprimido. En nuestro caso está ubicado en /xten-xlite/, nos ubicamos en esta ruta con ***cd /xten-xlite/*** y lo abrimos con el siguiente comando ***./xtensoftphone.***

Una vez abierto el programa procedemos a realizar la configuración dando clic en el iono *Menú* luego procedemos a dar clic en *System Settings* luego en *Sip Proxy* y por ultimo en *[Default]:* donde configuraremos los parámetros con los siguientes datos:

Enabled: Yes

Display Name: 201

Username:201

Authorization User: 201

Password:****

Domain/real:192.168.1.6

Sip Proxy: 192.168.1.6

Enabled: Yes

Display Name: 202

Username:202

Authorization User: 202

Password:****

Domain/real:192.168.1.6

Sip Proxy: 192.168.1.6

Una vez configurado estos parámetros damos clic en back para guardar los cambios, y cerramos la ventana de menú, luego de esto el softphone intentara autenticarse con el servidor asterisk, y de ser exitoso quedara listo para realizarlas llamadas.

Fig. 16 Softphone Registrado

3.4.3.4 Instalación y Configuración Softphone ZOIPER

Para instalar Zoiper, descargamos el archivo Zoiper Installer.exe; el asistente de instalación se abrir y damos clic en Siguiente; y luego en Aceptar.

Seleccionamos el icono en el escritorio y ejecutamos la instalación.

Una vez inicializado Zoiper procedemos a su configuración, en Sip accounts añadimos una nueva cuenta SIP con el nombre de G y lo configuramos como muestra la figura

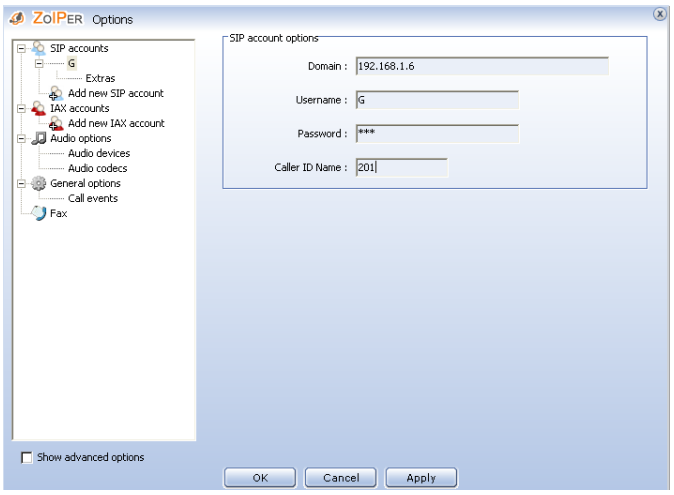

Fig. 17 Configuración de Zoiper

Luego de realizada la configuración el softphone intentara autenticarse con el servidor asterisk, y de ser exitoso quedara listo para realizarlas llamadas.

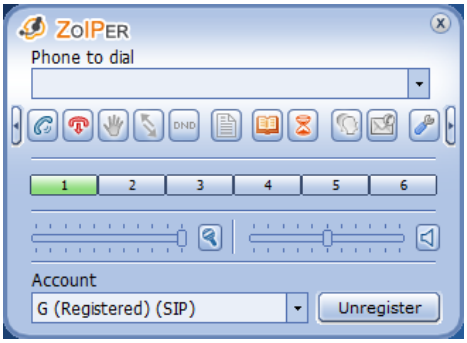

Fig. 18 Softphone Zoiper registrado

CAPITULO 4

PRUEBAS

4.1 INTRODUCCION:

Finalmente, se recogen una serie de pruebas que permiten verificar empíricamente el correcto funcionamiento del balanceo de las conexiones a servicios ofertados por el clúster. Estas mismas pruebas, permiten constatar que se aplica correctamente la persistencia de sesión para el servicio de asterisk.

La monitorización de contenidos también ha sido objeto chequeos, que comprueban su eficacia y correcto funcionamiento. Se han realizado simulacros de escenarios de fallida, que verificaron la alta disponibilidad del servicio, así como la información que maneja el servicio, se garantiza la plena disposición y replicación de los datos en diferentes escenarios de falla.

4.2 PLAN DE PRUEBAS

4.2.1 Pruebas de Sistema

Tabla IV Plan de Pruebas

No.	Pruebas	Duración	Responsable
1	**Pruebas de Conectividad**	**27 minutos**	**Rocha**
1.1	Verificación Ping a IP de maquinas del clúster.	15 minutos	**Rocha**
1.2	Verificación de resolución de nombres de servidor virtual. (nslookup)	10 minutos	**Rocha**
1.3	Verificación conexión servicio 'telnet'	2 minutos	**Rocha**
2	**Pruebas de Alta Disponibilidad**	**140 minutos**	**Castillo/Rocha**
2.1	Comprobando DRBD y Heartbeat	15 minutos	**Castillo**
2.2	Desconexión de servidor Asterisk1 (Principal)	5 minutos	**Castillo**
2.3	Desconexión de servidor Asterisk2 (Secundario)	5 minutos	**Castillo**
2.4	Apagado de servidor Asterisk1	10 minutos	**Castillo**
2.5	Apagado de servidor Asterisk2	10 minutos	**Rocha**
2.6	Desconexión de suministro eléctrico servidor Asterisk1	25 minutos	**Rocha**
2.7	Desconexión de suministro eléctrico servidor Asterisk2	25 minutos	**Rocha**
2.8	Desconexión general de suministro eléctrico	45 minutos	**Rocha**

4.3 DETALLE DE PRUEBAS

4.3.1 Pruebas de Conectividad

- Verificación Ping a IP de maquinas del clúster.

Tabla V Prueba de Ping

Ping	El usuario deberá realizar un ping a cada una de las direcciones IP. Ejemplo: 192.168.1.6
Resultado	Como resultado de esta prueba la conectividad será exitosa.
Posibles Errores	Servidor no responde frente a solicitud de ping.

- Verificación de resolución de nombres de servidor virtual. (nslookup)

Tabla VI Prueba de nslookup

Nslookup	El usuario deberá escribir el siguiente comando: asterisk2# nslookup –sil asterisk1
Resultado	El servicio de forma automática debe mostrar. Name: asterisk1 Address: 192.168.1.6
Posibles Errores	Nombre no resuelto.

- Verificación conexión servicio ‘telnet’

Tabla VII Verificación de servicio telnet

telnet	El usuario deberá escribir el siguiente comando: telnet a la dirección del servidor en donde está el servicio
Resultado	El servicio en forma automática debe mostrar. Conectándose a 192.168.1.3… Login: Password:
Posibles Errores	Conectándose a 192.168.1.3… No se puede abrir una conexión al host: Error en la conexión Maquina no existe

4.3.2 Pruebas de Alta Disponibilidad

- Comprobando DRBD y Heartbeat

Arrancamos DRBD tanto en Asterisk1 como en Asterisk2, una vez iniciado comprobamos que DRBD haya iniciado bien:

```
[root@asterisk1 ~]# service drbd start
Starting DRBD resources:    [ s(r0) ].
[root@asterisk1 ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.2.6 (api:88/proto:86-88)
GIT-hash: 3e69822d3bb4920a8c1bfdf7d647169eba7d2eb4 build by buildsvn@c5-i386-bui
ld, 2008-10-03 11:42:32
m:res  cs         st                ds                 p  mounted   fstype
0:r0   Connected  Primary/Secondary  UpToDate/UpToDate  C  /replica  ext3


[root@asterisk2 ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.2.6 (api:88/proto:86-88)
GIT-hash: 3e69822d3bb4920a8c1bfdf7d647169eba7d2eb4 build by buildsvn@c5-i386-bui
ld, 2008-10-03 11:42:32
m:res  cs         st                ds                 p  mounted  fstype
0:r0   Connected  Secondary/Primary  UpToDate/UpToDate  C
```

Fig. 19 Estatus DRBD en Asterisk 1 y 2

Esto nos indica que todo ha sido arrancado correctamente y que una sincronización completa está en marcha.

Arrancamos Heartbeat tanto en Asterisk 1 como en Asterisk2, una vez iniciado comprobamos que Heartbeat haya iniciado bien:

```
[root@asterisk1 ~]# service heartbeat status
heartbeat OK [pid 2062 et al] is running on asterisk1 [asterisk1]...

[root@asterisk2 ~]# service heartbeat status
heartbeat OK [pid 26584 et al] is running on asterisk2 [asterisk2]...
```

Fig. 20 Estatus de Heartbeat en Asterisk 1 y 2

- Comprobando que se puedan realizar llamadas.

Realizamos una llamada entre las extensiones y comprobamos que las llamadas se registren en el servidor:

```
asterisk1*CLI> sip show channels
Peer             User/ANR         Call ID          Format           Hold     Las
t Message    Expiry
0 active SIP dialogs
  == Using SIP RTP CoS mark 5
    -- Executing [202@internal:1] Dial("SIP/201-00000000", "SIP/202,10") in new
stack
  == Using SIP RTP CoS mark 5
    -- Called 202
    -- SIP/202-00000001 is ringing
```

Fig. 21 Sip show channels

Una vez montada la partición /dev/drbd0 en /replica en el servidor primario, copiamos algo a ella:

- `# cp /etc/hosts /replica/`

Desmontamos el recurso y hacemos que el primario pase a secundario:

- `# umount /replica`
- `# drbdadm secondary r0`

Si ahora ejecutamos un ‘service drbd status’ veríamos:

- `Secondary/Secondary`

Nos vamos al secundario y lo volvemos primario montamos la partición y comprobamos que este el archivo:

- `drbdadm primary r0`
- `mkdir /replica`
- `mount -t ext4 /dev/drbd0 /replica`
- `ls /replica`

En este punto el primario (asterisk1) ha tomado el rol secundario y el secundario (asterisk2) ha tomado el rol del primario.

Realizamos una llamada entre las extensiones configuradas para comprobar su correcto funcionamiento:

Fig. 22 Llamada de Prueba

- Desconexión de servidor Asterisk1 (Principal)

Esta prueba corresponde a desconectar el cable de red en el servidor principal y así forzar una falla:

Resultados:

Antes de desconectar Asterisk1 que es el servidor que está dando el servicio de Asterisk, revisamos el estado de este servicio en ambos servidores:

```
[root@asterisk1 ~]# service asterisk status
Se está ejecutando asterisk (pid 2536)...
[root@asterisk2 ~]# service asterisk status
asterisk está parado
```

Fig. 23 Estatus de asterisk en servidores

Procedemos con la desconexión.

El servidor secundario iniciara la secuencia de traspaso por falla al detectar que su homologo no responde, es decir el servidor que se encuentra activo cambia su estado a inactivo y el servidor secundario pasa al modo activo, tomando el control del servicio y permitiendo la continuidad de la operación de este. Veremos que la partición es montada automáticamente debido a Heartbeat.

```
[root@asterisk2 ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.2.6 (api:88/proto:86-88)
GIT-hash: 3e69822d3bb4920a8c1bfdf7d647169eba7d2eb4 build by buildsvn@c5-i386-bui
ld, 2008-10-03 11:42:32
m:res  cs            st               ds                 p  mounted   fstype
0:r0   WFConnection  Primary/Unknown  UpToDate/DUnknown  C  /replica  ext3
```

Fig. 24 Service drbd status en Asterisk 2

Ahora revisamos el estado del servicio en ambos servidores, dándonos como resultado que el servicio se levanta automáticamente en Asterisk2 demostrando que Heartbeat funciona correctamente:

```
[root@asterisk2 ~]# service asterisk status
Se está ejecutando asterisk (pid 27511)...

[root@asterisk1 ~]# service asterisk status
asterisk está parado
```

Fig. 25 Comprobacion del servicio de Asterisk

Errores:

La secuencia de traspaso por falla se ejecuta, por consiguiente el servidor deja de prestar los servicios a sus clientes.

- Desconexión de servidor Asterisk2 (Secundario)

Esta prueba contempla realizar la operación inversa a la anterior, es decir, se debe forzar una falla en el servidor secundario (activo), previa conexión del servidor antes desconectado. Y así provocar que la secuencia por falla se produzca a la inversa.

Resultados:

El servidor principal iniciara la secuencia de traspaso por falla al detectar que su homologo no responde, es decir el servidor que se encuentra activo cambia su estado a inactivo y el servidor primario pasa al modo activo, tomando el control del servicio y permitiendo la continuidad de la operación de este.

```
[root@asterisk1 ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.2.6 (api:88/proto:86-88)
GIT-hash: 3e69822d3bb4920a8c1bfdf7d647169eba7d2eb4 build by buildsvn@c5-i386-bui
ld, 2008-10-03 11:42:32
m:res  cs            st               ds                 p  mounted   fstype
0:r0   WFConnection  Primary/Unknown  UpToDate/DUnknown  C  /replica  ext3
```

Fig. 26 Service drbd status en Asterisk 1

De la misma manera que la prueba anterior, verificamos que el Heartbeat sigue funcionando a pesar de la desconexión, la partición es montada automáticamente y el Asterisk1 vuelve a tomar el control del servicio Asterisk.

Errores:

La secuencia de traspaso por falla no se ejecuta, por consiguiente el servidor deja de prestar los servicios a sus clientes.

- Apagado de servidor Asterisk1

Apagar el servidor principal normalmente.

- `asterisk1# shutdown –h now`

Resultados:

El servidor secundario detectara la ausencia del servidor principal (activo) e iniciara la secuencia de traspaso por falla, tomando el control del servicio.

Copiamos unos archivos más a /replica:

- `# cp /etc/hosts.* /etc/group /replica`

Y encendemos otra vez asterisk1. Cuando encienda empieza a sincronizar la partición drbd. Es decir los datos están replicados en los dos, pero solo puede acceder y montar el recurso para escribir uno de ellos en todo momento.

Una vez ha arrancado el secundario (asterisk1) volvemos a chequear el primario y veremos que vuelve a estar sincronizado:

```
[root@asterisk2 ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.2.6 (api:88/proto:86-88)
GIT-hash: 3e69822d3bb4920a8c1bfdf7d647169eba7d2eb4 build by buildsvn@c5-i386-bui
ld, 2008-10-03 11:42:32
m:res  cs         st                ds                 p  mounted   fstype
0:r0   Connected  Primary/Secondary  UpToDate/UpToDate  C  /replica  ext3
```

Fig. 27Service drbd status en Asterisk 2

Errores:

El servidor secundario no detecta la ausencia y por consiguiente no se realiza el traspaso por falla, lo cual dejara sin servicio, provocando la interrupción de este.

- Apagado de servidor Asterisk2

Antes de realizar esta prueba debe iniciarse el servidor apagado en la prueba anterior. Una vez restablecido se debe proceder al apagado del servidor secundario (activo).

Resultados:

El servidor (inactivo) detectara la ausencia del servidor secundario activo, por lo tanto se iniciara la secuencia de traspaso por falla, tomando este el control del servicio.

Errores:

El servidor primario (inactivo) no detecta la ausencia del servidor secundario activo, y por consiguiente no se realiza el traspaso por falla, lo cual interrumpirá la continuidad del servicio.

- Desconexión de suministro eléctrico servidor Asterisk1

Se debe desconectar el suministro de energía al servidor principal activo, lo que provocara el apagado anormal de este. (caída de sistema).

Resultados:

El servidor secundario notara la ausencia de su homologo e iniciara la secuencia de traspaso por falla, tomando el control del servicio como servidor activo.

Además el servidor principal caído presentara los errores respectivos, por lo cual se iniciaran los procedimientos de recuperación y

aseguramiento de la integridad del sistema de archivos, una vez iniciado el servidor deberá retomar su rol de activo, además de informar a su secundario que debe desactivarse.

Errores:

Existen 2 posibles errores en este caso:

El servidor secundario no detecta la ausencia y no se inicia la secuencia de traspaso por falla, provocando la interrupción del sistema por no haber un servidor que gestione las peticiones y respuestas.

El servidor caído no inicia correctamente debido a una falla mayor irrecuperable (ejemplo: daño físico o lógico), por lo cual no podrá retomar su rol.

- Desconexión y reconexión general de suministro eléctrico

Provocar la caída simultanea de todas las maquinas que componen el clúster, las cuales se reiniciaran de manera anormal.

Resultados:

Todas las maquinas deben iniciar sus procedimientos de recuperación y regresar paulatinamente a prestar sus servicios al servidor virtual, una vez que se encuentre a lo menos 1 servidor activo, este debería reanudar la prestación de servicios.

Apagamos ambos servidores y encendemos el primario (asterisk1). Veremos que monta la partición drbd y la IP virtual este accesible (la podemos ver haciendo un ifconfig).

También comprobamos que es el primario:

- `# service drbd status`
- `veríamos:`
- `Primary/Unknown`

A continuación arrancamos el secundario (asterisk2). Y comprobamos que es el secundario:

- `# service drbd status`
- `Secondary/Primary`

Y que sincroniza:

```
[root@asterisk2 ~]# service drbd status
drbd driver loaded OK; device status:
version: 8.2.6 (api:88/proto:86-88)
GIT-hash: 3e69822d3bb4920a8c1bfdf7d647169eba7d2eb4 build by buildsvn@c5-i386-bui
ld, 2008-10-03 11:42:32
m:res  cs         st                ds                p  mounted  fstype
0:r0   Connected  Secondary/Primary  UpToDate/UpToDate  C
```

Fig. 28 Service drbd status en Asterisk 2

Errores:

Solo un servidor reinicia correctamente, provocara que el servidor pierda su característica de alta disponibilidad pues no contara con un respaldo frente a una posible falla de sí.

Ningún servidor reinicia correctamente, el servidor virtual no podrá responder a las solicitudes recibidas pues no habrá quien las procese.

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

1. En en una infraestructura donde los fallos son importantes porque se deja de dar un servicio, mantenerlos corriendo en varias maquinas de forma redundante ayuda a evitar estas situaciones.

2. Heartbeat es la mejor herramienta para dar alta disponibilidad a un servicio, en este caso Asterisk; ya que siendo un software libre se adapta a las necesidades del cliente y proporciona un clúster evitando en lo más mínimo la pérdida del servicio de telefonía

3. La instalación y configuración de Heartbeat son procedimientos muy complejos pero necesarios para tener una buena estructura de clustering.

4. El uso de DRBD junto con Heartbeat optimiza en un 100% el desempeño de la solución; puesto que se crea una partición virtual la misma que será compartida entre los servidores que se realice el clustering

5. Con Heartbeat, la adquisición de la IP se produce en menos de un segundo. DRBD se re-configura en casi 30 segundos y luego dependiendo de la plataforma de hardware y la complejidad de las aplicaciones que se ejecuten en Asterisk puede tardar entre 5-15 segundos para que Asterisk se ponga en marcha en el servidor secundario, se sincronizan los archivos de configuración y esta listo para procesar las llamadas. Tiempo total del fail-over es entre 15-20 segundos.

6. No toda la información que existe en la Internet acerca de la instalación y configuración es correcta ya que existen varios procedimientos erróneos.

Recomendaciones:

1. No confiarse en información que se encuentra en la Internet ya la mayoría de procedimientos son erróneos.

2. Es necesario chequear: nombres de host y direcciones IP, ya que son datos importantes para el correcto funcionamiento de DRBD y Heartbeat.

3. Cuando se usa DRBD es posible que se presente un problema conocido como “Split Brain”, se deberá consultar [2] para tener conocimiento de cómo solucionarlo contrario la sincronización no se completara.

Bibliografía

[1] Telesoft Integrando Technologies, Building Elastix-1.3 High Availability Clusters with Redfone foneBRIDGE2, DRBD and Heartbeat, http://support.red-fone.com/downloads/elastix/Elastix_HA_Cluster.pdf, 23 de Diciembre de 2008

[2] Cancino Marcos, Cluster de Alta Disponibilidad y Bajo Costo Cluster de Comunicaciones Sobre Linux, http://es.scribd.com/doc/36757812/Cluster-de-Bajo-Costo-en-Linux, Octubre 2002

[3] Marcote Gonzalo, HA Asterisk – Alta disponibilidad Asterisk (I). Heartbeat + Drbd, http://www.gonzalomarcote.com/blog/?p=58, Noviembre 2009

[4] Sandiford Bill, Asterisk Redundancy Using Heartbeat, http://taug.ca/files/TAUG-Heartbeat.pdf, 28 de Febrero de 2008