Universidad Austral de Chile

Facultad de Ciencias de la Ingeniería

Escuela de Ingeniería Civil en Informática

"INTEGRACÍON DE APLICACIONES DE SOFTWARE LIBRE APLICADO A UN CLUSTER DE ALTA DISPONIBILIDAD Y BALANCEO DE CARGA DE SERVIDORES PROXY."

Tesis para optar al Título de:
Ingeniero Civil en Informática.

Profesor Patrocinante:
Sr. Raimundo Ezequiel Vega Vega.
Estadístico.
Master en Informática.
Doctor en Informática.

Profesor Co Patrocinante.
Sr. Erick Alexis Araya Araya.
Ingeniero de Ejecución Electrónico
Magíster en Ingeniería Electrónica.

DANIEL ANTONIO EUGENIN MORALES

Valdivia – Chile

2005

Universidad Austral de Chile

Instituto de Informática

Valdivia 5 de diciembre 2005

Sra.
Miguelina Vega Rosales
Directora Escuela de Ing. Civil en Informática

De mi consideración,

Informo a usted que he revisado el trabajo de tesis titulado, "Integración de aplicaciones de Software Libre Aplicado a un Cluster de Alta Disponibilidad y Balanceo de Carga de Servidores Proxy", realizado por el Sr. Daniel Eugenin Morales, egresado de la Escuela de Ingeniería Civil en Informática.

La estructura de la tesis del Sr. Eugenin permite comprender claramente los objetivos de su trabajo y como estos son logrados. Las tecnologías asociadas al diseño e implementación de la solución propuesta al problema planteado, son examinadas in extenso, seleccionando las adecuadas para la solución. La integración de tecnología mostrada en la Tesis, dan cuenta de la expertitud que el Sr. Eugenin ha alcanzado en la construcción de soluciones de infraestructura y software básico.

Basado en lo anterior, he resuelto calificar el trabajo del Sr. Eugenin con nota 7.0 (siete).

Reciba un cordial saludo,

Dr. Raimundo Vega Vega
Instituto de Informática
Facultad de Ciencias de la Ingeniería

Dirección : General Lagos 2086 - Campus Miraflores- Valdivia - Chile

Fono: 56 63 221427 Fax: 56 63 293115 email: instituto@inf.uach.cl

COMUNICACIÓN INTERNA N°/05 REF. CALIFICACIÓN PROYECTO DE TÍTULO.

VALDIVIA, 15 de Diciembre de 2005

DE : ERICK ARAYA A.

A : DIRECTORA ESCUELA INGENIERÍA CIVIL EN INFORMÁTICA

MOTIVO:

INFORME TRABAJO DE TITULACIÓN

Nombre Trabajo de Titulación: INTEGRACIÓN DE APLICACIONES DE SOFTWARE LIBRE APLICADO A UN CLUSTER DE ALTA DISPONIBILIDAD Y BALANCEO DE CARGA DE SERVIDORES PROXY

Nombre del Alumno: DANIEL ANTONIO EUGENIN MORALES

Nota; 6,0 SEIS COMA CERO
(en números) (en letras)

FUNDAMENTO DE LA NOTA:
El trabajo si bien resuelve un problema real, de modo aparentemente eficiente, no logra concretar en la escritura la complejidad implícita. El análisis de la solución se observa liviano, con pocos argumentos para decidir la validez de los productos examinados. Tampoco se observa mayor análisis de software en el ambiente Windows. ¿No los hay? Respecto a código, se muestra sólo lo relacionado a instalación y configuración... Supongo habrá aportes del alumno en tan interesante solución.

Considerar:
- Cumplimiento del objetivo propuesto
- Satisfacción de alguna necesidad
- Aplicación del método científico
- Interpretación de los datos y obtención de conclusiones
- Originalidad
- Aplicación de criterios de análisis y diseño
- Perspectivas del trabajo
- Coherencia y rigurosidad lógica
- Precisión del lenguaje técnico en la exposición, composición, redacción e ilustración.

Atentamente,

ERICK ARAYA A.

Universidad Austral de Chile

Instituto de Informática

Valdivia, 15 de diciembre del 2005.

A: Prof. Miguelina Vega. Directora Escuela de Ing. Civil en Informática.
De: Luis A. Álvarez G.

Motivo.
Informar calificación del siguiente Trabajo de Titulación:

Título:
INTEGRACIÓN DE APLICACIONES DE SOFTWARE LIBRE APLICADO A UN CLUSTER DE ALTA DISPONIBILIDAD Y BALANCEO DE CARGA DE SERVIDORES PROXY

Alumno:
SR. DANIEL ANTONIO EUGENIN MORALES.

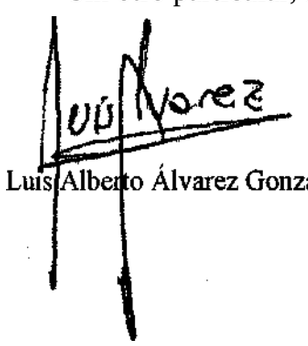
Nota:
SEIS COMA CERO (6,0)

JUSTIFICACIÓN DE LA NOTA:
Tiene el mérito de un trabajo de muy alto nivel técnico, donde demuestra su gran conocimiento en software libre y su capacidad en integrar diferentes aplicaciones, en particular lo referido a la construcción de cluster de carga balanceada.

Sin embargo, a su trabajo le falta mayor sustento teórico; rápidamente se va a las especificaciones técnicas; lo cual además se ve reflejado en las referencias bibliográficas, la mayoría de ellas corresponden a textos en Web, al estilo manual de procedimientos. Aquí el alumno comete un leve error al colocar el año en que fueron consultadas (en su mayoría 2005) con el año en que fueron publicadas.

Ajuicio del evaluador, dado el trabajo desarrollado, el alumno posee más méritos que los que se pueden apreciar en el trabajo escrito.

Sin otro particular, se despide atentamente



Luis Alberto Álvarez González

AGRADECIMIENTOS

Quiero agradeder a Rosario por acompañarme y darme su apoyo incondicional en todo el proceso de escritura de esta Tesis.

Tambien a mis amigos y familia quienes me incentivaron a esforzarme en la realización de este trabajo.

Finalmente, mi especial agradecimiento a mis padres, Hector y Zunilda, quienes me han dado todo lo que soy ahora; por su apoyo y su comprensión durante toda mi etapa estudiantil.

A todos ellos, les dedico este trabajo.

Gracias a todos.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

RESUMEN

Un servidor Proxy tiene como finalidad compartir una sola conexión a Internet a varios usuarios al mismo tiempo. Las máquinas de la red realizan sus peticiones al servidor Proxy, éste se conecta con el sitio especificado, y los datos devueltos por Internet los envía a la máquina de la red que hizo la petición. Tiene la posibilidad de almacenar (caching) las páginas Web que ya han sido visitadas con el fin de reducir el tiempo de acceso y el uso de ancho de banda, además de guardar registros de navegación por cada usuario que hay dentro de una compañía.

El servicio de un Proxy, entonces, juega un papel muy importante dentro de una organización, y al mismo tiempo se transforma en un servicio crítico al ser utilizado por la gran mayoría de los usuarios para acceder a Internet, siendo necesario la implementación de un sistema de cluster que permita mantener el servicio disponible en caso de que falle un servidor, tal es el caso de los cluster de alta disponibilidad y de balanceo de carga.

La realización de este proyecto nació por la necesidad de una compañía de poseer un sistema cluster de servidores Proxy, para el cual se analizaron diferentes soluciones -comerciales y no comerciales- y, dependiendo de los requerimientos que tenía la compañía, se eligió el sistema de cluster más adecuado: una solución libre que implementa un cluster de alta disponibilidad y de balanceo de carga.

Además de las complejidades técnicas que pudiera tener la instalación, configuración y tunning de una solución de cluster de servidores Proxy, la compañía tenía un requerimiento extra: la utilización del servidor de directorio (Active Directory) de la organización para realizar la autentificación y la autorización de los usuarios que pueden navegar por Internet, lo cual hacía más complejo el desarrollo de la solución, obligando al autor realizar un estudio profundo de las principales tecnologías del mundo Open Source que, integrándolas todas ellas, formarían una solución confiable, robusta y estable que actualmente se encuentra en producción.

ABSTRACT

A Proxy server is a device used to share an Internet conection for several users. The machines of the network send their requests to the Proxy server, the server connects to the requested site, and the resulting data is then given by the server back to the client that made the request. It also has the possibility of caching the Web pages that already have been visited to reduce the access time and bandwidth usage. A proxy server will also keep accounting data for each client.

The Proxy service plays a very important role within an organization, easily making itself a critical service used by the great majority of the users to access to Internet, normally needing the implementation of a cluster system that allows maintaining the service available in the event of a server failure. In this case, the cluster is a high availability and load balance cluster.

This proyect was born out of necessity of a company to have clustered Proxy servers, for which different solutions -commercial and noncommercial- were analyzed and, according to the requirements that the company had, a free solution that implemented a high availability and load balancing cluster was selected.

In addition to the technical complexities that the installation, configuration and tunning of a clustered Proxy server solution, the company had an extra requirement: the use of an Active Directory server already existant in the organization to authenticate and authorize users who could navigate the Internet, which made the solution much more complex and forcing to the author to deeply learn several Open Source technologies that, integrated, could make a reliable, robust and stable production solution.

CAPÍTULO 1

INTRODUCCIÓN

La red mundial ***Internet*** se ha convertido en una herramienta necesaria para la mayoría de las organizaciones actuales, utilizándola para buscar información, entretenimiento, noticias, servicios, etc. Por esta razón es que las conexiones a Internet (en cuando a velocidad) y las restricciones de acceso juegan un papel muy importante en el momento en que una organización desea acceder a ella. Esto lleva a las organizaciones a buscar alguna solución que se adapte a sus necesidades y que cumpla con requisitos, tales como restricciones de acceso, mantención de cache, registros de navegación, etc., siendo el servidor ***Proxy*** uno de los servicios que entra a jugar un papel muy importante en este ámbito.

Un servidor Proxy tiene como finalidad compartir una sola conexión a Internet a varios usuarios al mismo tiempo. Las máquinas de la red realizan sus peticiones al servidor Proxy, éste se conecta con el sitio especificado, y los datos devueltos por Internet los envía a la máquina de la red que hizo la petición. Tiene la posibilidad de almacenar (caching) las páginas Web que ya han sido visitadas con el fin de reducir el tiempo de acceso y el uso de ancho de banda.

Existen muchísimas aplicaciones -comerciales y no comerciales- que se pueden utilizar para implementar un servidor Proxy. Cada una de estas aplicaciones posee sus propias características que ofrecen a las organizaciones diversas posibilidades a la hora de decidir cuál de estas aplicaciones utilizar. Una de estas aplicaciones es ***Squid***[1], la cual es una de las más populares utilizadas hoy en día alrededor del mundo gracias a su gran robustez, estabilidad y confiabilidad.

Dentro de las bondades de Squid, se puede señalar que es el servidor Proxy

1 Squid es el nombre del servicio que implementa un Proxy en Linux.

más popular y extendido en el mundo, es confiable, robusto y versátil. Al ser software libre, además de estar disponible el código fuente, está libre de pago de costosas licencias por uso o con restricción a un uso con determinado número de usuarios. Squid es ideal para acelerar el acceso a Internet, y para controlar el acceso a diversos sitios Web.

El servicio de un Proxy, entonces, juega un papel muy importante dentro de una organización, y al mismo tiempo se transforma en un servicio crítico al ser utilizado por la gran mayoría de los usuarios para acceder a Internet, siendo necesario la implementación de un sistema de ***cluster*** que permita mantener el servicio disponible en caso de que falle un servidor.

Al hablar de cluster, tenemos un numeroso listado de diversas aplicaciones que implementan distintos tipos de cluster, dependiendo de las necesidades que posea la organización y la aplicación a clusterizar.

Dentro de los cluster mas comunes, se encuentra el ***cluster de alta disponibilidad***, en el cual uno de los nodos actúa pasivamente mientras el nodo activo recibe todas las peticiones a los servicios que ofrece. En caso de que el nodo activo tenga alguna falla en los servicios, el nodo pasivo toma el control de los servicios y pasa a ser el activo para que los servicios ofrecidos estén siempre disponibles.

Actualmente, debido a la gran cantidad de usuarios que necesitan acceder a los servicios, es necesario también aprovechar los nodos pertenecientes al cluster, para que estos pasen a ser activos y la carga se pueda dividir entre todos los nodos del cluster, constituyendo así un ***cluster de balanceo de carga***.

El proyecto que se propone a continuación presenta una solución para una compañía que requiere un aprovechamiento del hardware existente, además de la continuidad del servicio Proxy en caso de que una máquina deje de funcionar.

Dos líneas principales posee el proyecto: la primera de ellas es la configuración de un servidor Proxy y su conexión con un servidor Microsoft® Active Directory®[2] para lograr la autorización de los usuarios que pueden navegar por Internet; mientras que la segunda línea de trabajo consta de la creación de un cluster de alta disponibilidad y de balanceo de carga del servidor Proxy utilizando 2 nodos.

1.1. Antecedentes existentes al respecto

De un tiempo a esta parte han ido surgiendo proyectos y soluciones para Linux, en las cuales algunas destacan su elegancia y sencillez en comparación con sistemas comerciales. Entre las soluciones HA (High Availability, o Alta Disponibilidad) para Linux, tenemos:

Heartbeat: Es una herramienta que permite crear un cluster de alta disponibilidad. De momento el cluster sólo soporta 2 nodos, permite crear grupos de recursos y cambiarlos fácilmente entre los nodos que lo conforma. Carece de herramientas de monitorización del servicio de datos. [Paredes, 2004].

Ultramonkey: Es una solución creada por VA Linux que se basa en LVS (Linux Virtual Servers) y Heartbeat para ofrecer cluster de alta disponibilidad y balanceo de carga, en el cual hay un nodo que se encarga de gestionar y repartir las conexiones (Nodo Master LVS) entre todos los nodos Slave del cluster. El servicio de datos debe residir en todos los nodos Slave. LVS puede llegar a soportar sin problemas hasta 200 nodos Slave. [Horman, 2004]

Red Hat Cluster Suite: es una solución comercial de Red Hat, Inc. para compañías que requieren de aplicaciones de alta disponibilidad, mayor rendimiento y disponibilidad de sus estructuras de red. Provee dos distintos tipos de clustering:

2 Servidor de Directorio creado por Microsoft® Corporation.

Cluster Manager, para proveer alta disponibilidad en aplicaciones que utilizan tecnología de failover; *IP Load Balancing*, para proveer la habilidad de balanceo de carga en una granja de servidores. [Red Hat, 2005]

OpenMosix: es una extensión del kernel de Linux para clustering de sistemas independientes. Esta extensión del kernel torna una red de computadores ordinarios en un supercomputador para aplicaciones de Linux. Una vez instalado OpenMosix, los nodos en el cluster comienzan a comunicarse uno con otro y el cluster se adapta a la carga de trabajo. Los procesos se originan desde cualquier nodo, si ese nodo está demasiado ocupado comparado a otros, el proceso puede emigrar a cualquier otro nodo. OpenMosix procura continuamente optimizar la asignación de recursos. [Moshe, 2004]

Kimberlite: Creada por Mission Critical Linux, es una solución que soporta un cluster de 2 nodos. Permite fácilmente, definir un dispositivo de quórum, monitorizar los servicios de datos, así como gestionarlo. Una solución completa bajo GPL. [Kimberlite, 2004]

1.2.Objetivos

1.2.1.Objetivos generales

Diseño y construcción de un cluster de alta disponibilidad y balanceo de carga de servidores Proxy basado en la integración de diversas aplicaciones de software libre.

1.2.2.Objetivos específicos

- Analizar y diseñar una solución de alta disponibilidad y balanceo de carga para servidores Proxy.

- Implementar la configuración de dos servidores Proxy.
- Construir un cluster de alta disponibilidad y de balanceo de carga.
- Integrar herramientas de reporte y monitoreo que informe el estado de cada servidor Proxy.
- Implementar una solución para el respaldo y restauración parcial o total del sistema.

1.3.Estructura de esta Tesis

Capítulo 1. Breve **Introducción** al tema.

Capítulo 2. Se hace un recorrido por la **Tecnologías Asociadas** a este proyecto. Se cubren las principales tecnologías: Proxy, Active Directory, Samba, Kerberos y Cluster.

Capítulo 3. **Especificación de requisitos** por parte del cliente. Se documenta los requisitos del cliente en cuanto a sus necesidades, facilidad de uso, mantención del sistema, confiabilidad, escalabilidad, rendimiento, instalación y soporte operacional.

Capítulo 4. **Análisis y diseño de la solución propuesta**. En la cual se analizan diversas alternativas de solución y se selecciona una, para luego documentar la solución propuesta y preparar un plan inicial del proyecto.

Capítulo 5. **Implementación e instalación**, cubriendo las configuraciones de los servicios.

Capítulo 6. **Puesta en producción y soporte**.

Capítulo 7. **Conclusiones**.

CAPÍTULO 2

TECNOLOGÍAS ASOCIADAS

Este capítulo tiene por finalidad realizar un recorrido por las principales tecnologías con las cuales se trabajará en el desarrollo de esta Tesis. Cada una de estas tecnologías han sido elegidas por diversas razones que se detallarán en el capítulo 4.

2.1.PROXY (Squid)

Un servidor Proxy es un programa que crea y mantiene una conexión de red en beneficio de otro programa, proporcionando un conducto en el nivel de aplicación entre un cliente y un servidor. El cliente y el servidor, en realidad, no tienen comunicación directa. El Proxy aparenta ser el servidor para el programa cliente y parece ser el cliente para el programa servidor. Los programas cliente se conectan a un servidor Proxy en lugar de a un servidor remoto. El Proxy establece la conexión con el servidor remoto en beneficio de la aplicación cliente, después de sustituir la dirección origen del cliente con la suya. [Ziegler, R. 2000]

Squid es un servidor de Proxy caché de alto rendimiento para clientes Web, soportando objetos de datos FTP, Gopher y HTTP y que se utiliza muy a menudo en sistemas Linux. Este tipo de programas se utiliza para acelerar el acceso a la Web de los usuarios internos de una red, a la vez que registra los sitios que son visitados por estos usuarios. [Hatch, Lee y Kurtz. 2001]

Squid es el software para servidor Proxy más popular y extendido entre los sistemas operativos basados sobre UNIX®. Es muy confiable, robusto y versátil. Al ser software libre, además de estar disponible el código fuente, está libre de pago de

costosas licencias por uso o con restricción a un uso con determinado número de usuarios.

2.1.1.Características de Squid

Squid puede hacer Proxy y cache de los protocolos HTTP, FTP, Gopher, Proxy de SSL, cache transparente, aceleración HTTP, caché de consultas DNS y muchas otras cosas más, como filtración de contenidos y control de acceso por IP y por usuario. A través del uso del Protocolo Ligero de Caché de Internet (ICP), lo que Squid almacena pueda estar disponible en una jerarquía o una malla de servidores Proxy para almacenamientos adicionales de ancho de banda. Su misión es compartir una sola conexión a internet entre los computadores de la red local. Los computadores de la red realizan sus peticiones al servidor Proxy, éste se conecta con el sitio especificado, y los datos devueltos a través de Internet, los envía al computador de la red que hizo la petición. [Chadd, 2004]

La estructura de Squid está compuesta por un programa servidor principal, llamado *Squid*, un programa de resolución de Nombres de Dominio (DNS), *dnsserver*, algunos programas opcionales para reescribir peticiones y funciones de autenticación, y algunas herramientas de administración y clientes. Cuando Squid se inicia, éste genera un número configurable de procesos dnsserver, cada uno de ellos puede resolver un único y bloqueante petición DNS. Esto reduce la cantidad de tiempo que la petición DNS espera en la caché. [Chadd, 2004]

2.1.2.Funcionamiento de Squid

La estructura de funcionamiento de Squid es la siguiente:

Figura 1: Funcionamiento de Squid

0) En un inicio, el servidor Proxy (Squid) escucha las peticiones en el puerto por defecto (3128).
1) Cuando un cliente dentro de la red desea navegar por algún sitio Web, el sitio es solicitado al servidor Proxy.
2) El servidor Proxy procederá a navegar por Internet en búsqueda de dicha página solicitada por el cliente.
3) La página encontrada será devuelta al servidor Proxy.
4) El servidor Proxy guardará una copia de la página solicitada en una caché del disco duro.
5) El servidor Proxy, finalmente, le entregará al cliente la página solicitada.

2.1.3.Configuración de Squid

La configuración de un servidor Proxy Squid consta de muchísimos parámetros de los cuales se extraerán los principales -para poder poner en marcha el servidor- y que se detallan a continuación: [Jara, 2005]

http_port (asignar puerto)

Permite especificar uno o varios puertos de escucha para el servidor Squid. Si no se especifica este parámetro por defecto Squid escuchará en el puerto 3128. Sus valores pueden ser: [dirección IP:] puerto

Ejemplos:

http_port 3128 8080

http_port 172.26.192.68:3128

cache_mem (tamaño para la caché en memoria)

Permite indicar la cantidad ideal de memoria RAM -como máximo- para almacenar caché de objetos en tránsito, objetos en caliente y objetos almacenados en la caché. Los datos de estos objetos se almacenan en bloques de 4KB. Cache_mem especifica un límite ideal en el tamaño total de bloques acomodados, donde los objetos en tránsito tienen mayor prioridad, es decir que el resto de objetos la podrán usar hasta que sea requerida. En el supuesto caso que el objeto en tránsito requiera una memoria mayor a la especificada se excederá para satisfacer la petición.

Ejemplo:

cache_mem 16 MB

Si el servidor tiene como mínimo 128 MB de RAM es aconsejable indicar este valor.

cache_dir (tamaño y directorio para la caché física)

Permite indicar la cantidad de memoria física máxima para almacenar caché en el disco duro, es decir cuanto espacio almacenar de objetos de Internet. Sus valores pueden ser: tipo directorio tamaño número_subdirectorios número_niveles.

Ejemplo:

cache_dir ufs /var/spool/squid 100 16 256

El numero 100 corresponde a 100MB como espacio máximo para almacenar

caché, el 16 son el número de subdirectorios que contendrá el directorio principal (en este caso /var/spool/squid) y el 256 significa el número de niveles para cada subdirectorio. En caso de especificar un tamaño máximo inferior al espacio real disponible, el servidor Squid se bloqueará.

maximum_object_size (tamaño máximo para cacheados)

Permite especificar en kilobytes el tamaño máximo de los objetos que se pueden cachear, es decir, los objetos más grandes del tamaño indicado no serán cacheados.

Ejemplo:

maximum_object_size 4096 KB

En el ejemplo se han indicado 4MB como tamaño máximo de objetos en la caché.

cache_access_log (log de peticiones y uso de la caché)

Permite definir en que archivo Squid debe guardar una lista de las peticiones que va recibiendo, con la información de la página que se ha consultado y si ésta ha sido facilitada desde la caché o desde el servidor Web.

Ejemplo:

cache_access_log /var/log/squid/access.log

Squid presenta más directivas para definir donde registrar sus logs, a continuación se muestran las siguientes dos que pueden ser de mayor utilidad.

cache_log /var/log/squid/cache.log

Información general sobre el comportamiento de la caché

cache_store_log /var/log/squid/store.log

Muestra que objetos son ejecutados des de la caché y hasta cuando serán guardados.

reference_age (tiempo máximo en la caché)

Permite especificar el tiempo máximo que puede permanecer un objeto en la caché sin que se acceda a él, transcurrido ese tiempo será borrado. En Squid, el objeto que ha estado mas tiempo sin ser accedido lo calcula dinámicamente con el fin de ser borrado según el espacio disponible en la caché. Sus valores pueden ser: time-units, tal como se muestra a continuación.

reference_age 1 year

reference_age 3.5 days

referense_age 2 hours

refresh_pattern (factor para páginas sin caducidad)

Permite especificar que fecha en minutos deben de tener los documentos que su servidor no estableció una cabecera Expires indicando su caducidad.

Sus valores pueden ser: expresión_regular mín porcentaje máx, tal como se muestra a continuación.

refresh_pattern -i \.gif$ 14400 70% 43200

refresh_pattern ^ftp: 1440 20% 10080

refresh_pattern . 0 20% 4320

El valor correspondiente a la expresión regular debe de contener la especificación del objeto basándose en la dirección (URL) de la petición o un "." para indicar el resto. En los ejemplos se muestra como especificar cualquier objeto "gif" y cualquier petición "FTP".

El valor *mín* corresponde a los minutos mínimos que un objeto que no dispone de una cabecera Expires (indicando su caducidad), pueda ser considerado como no caducado (fresco). El valor "0" se recomienda para que no se obligue la retención de objetos no deseados como pueden ser los dinámicos.

El valor *porcentaje* sirve para especificar en aquellos objetos sin fecha de caducidad, cual será su fecha, aplicando un porcentaje sobre su tiempo desde la última modificación (la última modificación de un objeto se obtiene de la cabecera Last-Modified).

El valor *máx* corresponde a los minutos máximos que un objeto podrá ser considerado como no caducado.

ftp_user (acceso anónimo para FTP)

Permite indicar el correo que debe usarse como contraseña para acceder de forma anónima a un servidor FTP. Esto es útil si se desea acceder a servidores que validan la autenticidad de la dirección de correo especificada como contraseña.

Ejemplo:

ftp_user deugenin@Linuxcenterla.com

2.1.3.1.Autentificación y autorización de usuarios

2.1.3.1.1.Reglas de Control de Acceso

Squid posee una característica muy fuerte y flexible para controlar el acceso de usuarios. Para aplicarlos, se debe definir -en primer lugar- un conjunto de *Listas de Control de Acceso* (ACL) para luego aplicar las *Reglas de Control de Acceso* (http_access) para ellas.

Para aplicar una regla de control de acceso, se debe utilizar una combinación de una ACL definida y una declaración http_access para permitir o rechazar la navegación a ciertos usuarios.

Sintaxis:

acl <nombre> <tipo> <valores>

Ejemplos de tipos de ACL comúnmente utilizadas:

Tipo de ACL	**Referencia**
src	Dirección IP origen de cualquier conexión cliente
dst	Dirección IP de destino de cualquier conexión
srcdomain	Dominio origen de cliente (reverse lookup)
dstdomain	Dominio de destino de una URL
time	días/horas
url_regex	Expresiones regulares para URLs
proxy_auth	Utilizado para la autentificación de usuarios

2.1.3.1.2.Acceso a la navegación

Para controlar el acceso de la navegación de los usuarios, se utiliza el parámetro *http_access*, mediante la cual se puede definir si se desea permitir navegar a algún usuario, o hacia algún dominio, o en alguna hora determinada, etc.

Sintaxis:

```
http_access <allow|deny> [!] <nombre de la ACL>
```

Ejemplo:

```
acl mired src 172.26.0.0/16
http_access allow mired
```

En la primera línea se define la ACL llamada “mired”, la cual posee el valor

"172.26.0.0/16" correspondiente a toda una red de clase B. En la segunda línea es donde se permite la navegación a todas las máquinas que pertenezcan al valor que tiene la ACL "mired".

De esta misma forma se puede permitir o rechazar no solo direcciones IP de clientes, sino que se puede permitir o rechazar la conexión hacia algunos sitios (URLs), o dominios, o restringir el horario o los días de navegación, etc.

2.1.4.Configuración de un cliente para utilizar un servidor Proxy

2.1.4.1.Internet Explorer (IE)

Los computadores clientes deben configurarse para poder conectarse al servidor Proxy. Para ello, en Internet Explorer, en el menú *Herramientas* se debe seleccionar *Opciones de Internet.* En esta sección se debe elegir *Conexiones* (figura 2) y luego en la sección del Proxy elegir *Configuración* (figura 3)**.**

Figura 2: Opciones de Internet en IE

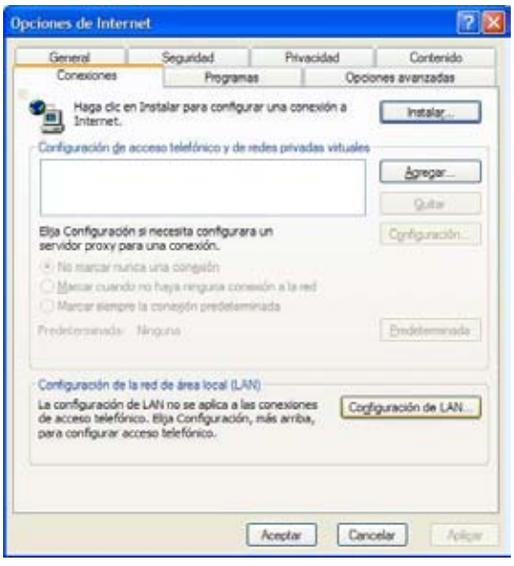


Figura 3: Configuración del Proxy en IE

En esta sección es donde se indica la dirección IP (o el Hostname) a la cual

accederá el cliente, que debe ser la misma que el servidor Proxy indicando además el puerto que se va a utilizar.

Figura 4: Configuración de los servidores Proxy por puerto en IE

2.1.4.2.Mozilla – Firefox

Los computadores clientes que posean un browser del tipo Mozilla/Firefox y que deseen utilizar un Proxy deben ir al menú principal y seleccionar *Editar*, para luego dirigirse a *Preferencias* (figura 5).

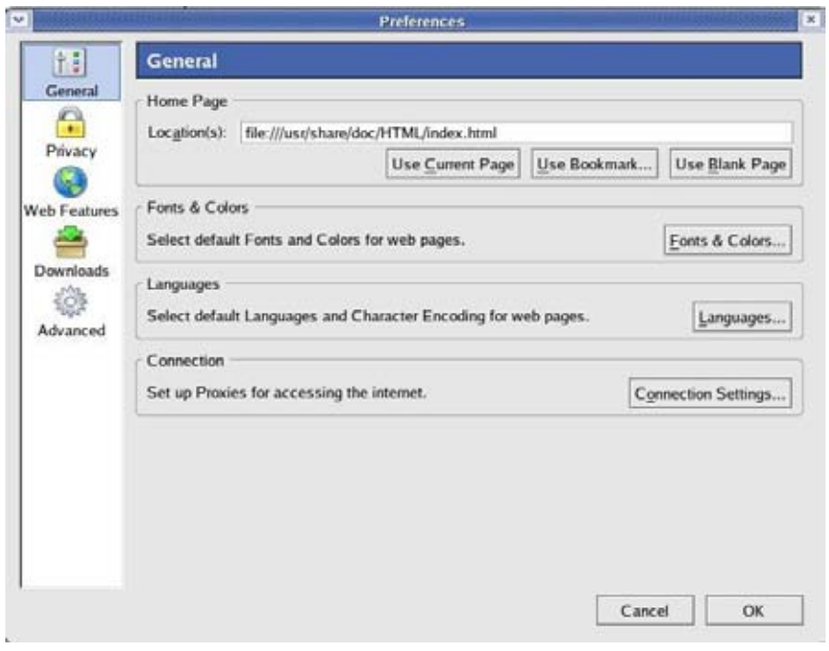


Figura 5: Preferencias de Internet en Firefox

En esta sección, debe seleccionarse la opción *Conections Settings*, y se debe escribir los parámetros de información del Proxy a utilizar (Hostname o IP y puerto que utiliza).

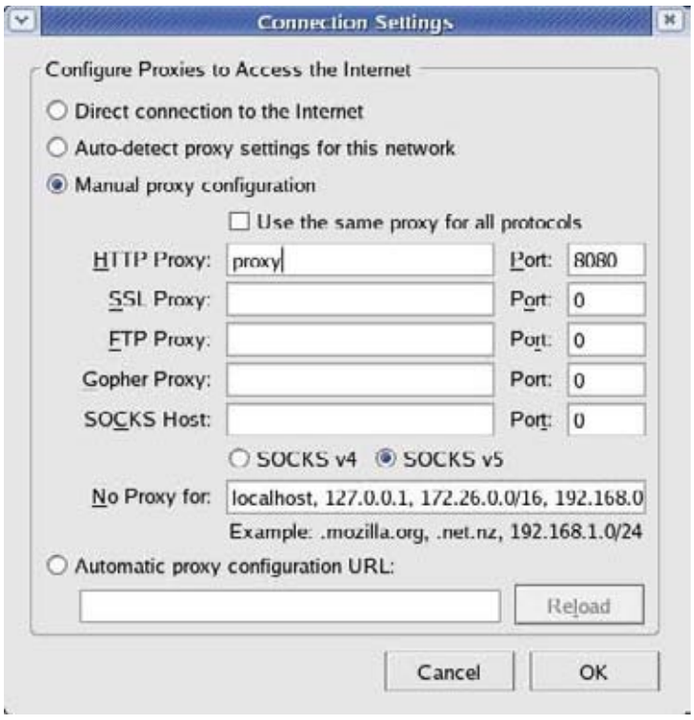


Figura 6: Configuración de Proxy por puerto en Firefox

2.2.ACTIVE DIRECTORY

Active Directory (AD) es un servicio de directorios. Un servicio de directorio es un servicio de red que identifica todos los recursos en ella y los vuelve accesible a los usuarios y a las aplicaciones. Active Directory es el servicio de directorio incluido en Microsoft® Windows® 2000/2003. [Microsoft, 2005]

El elemento principal de Active Directory es el *directorio*, que almacena información sobre los recursos de red y los servicios que hacen disponible la información. Los recursos almacenados en el directorio, como los datos del usuario, impresoras, servidores, bases de datos, grupos, computadores y políticas de sistema, se denominan objetos.

Active Directory organiza los objetos jerárquicamente en dominios. Un *dominio* es una agrupación lógica de servidores y otros recursos de red bajo un mismo nombre de dominio.

Cada dominio incluye uno o más controladores de dominio (domain controllers), que son máquinas que almacenan una réplica de un directorio de dominio. Cada vez que se hace algún cambio en alguno de los controladores, el resto se actualiza automáticamente.

Un *objeto* es un conjunto de atributos particulares, bajo un nombre específico, que representa un recurso individual de la red. Los *atributos* se refieren a las características del objeto. Así, los atributos de una cuenta de usuario pueden ser el nombre, departamento y dirección de email; y los de una impresora, si es láser y si es en color. Algunos objetos funcionan también como contenedores: por ejemplo, un dominio.

Las agrupaciones lógicas de objetos son las clases. Una clase puede estar constituida por todas las cuentas de usuario, las impresoras, los grupos, etc. [Microsoft. 2005]

Las *unidades organizacionales* (OU, Organizational Units) son contenedores que se usan para reunir objetos de un dominio en grupos administrativos lógicos. Cada OU puede contener distintos objetos y cada dominio puede tener su propia lógica de agrupación en OUs.

La unidad central de la estructura lógica de Active Directory es el dominio. Agrupando los objetos en uno o más dominios es posible representar la propia organización de la empresa. Todos los objetos de la red existen en un dominio, es posible albergar hasta 10 millones de objetos. [Microsoft. 2005]

Active Directory permite la administración de una red empresarial de manera fácil, centralizada y automática en muchos de sus parámetros.

2.2.1.Qué es un servicio de directorio

Un directorio es una base de datos optimizada para lectura, navegación y búsqueda. Los directorios tienden a contener información descriptiva basada en atributos y tienen capacidades de filtrado muy sofisticada. Los directorios generalmente no soportan transacciones complicadas ni esquemas de vuelta atrás como los que se encuentran en los sistemas de bases de datos diseñados para manejar grandes y complejos volúmenes de actualizaciones. Las actualizaciones de los directorios son normalmente cambios simples, o todo o nada, siempre y cuando estén permitidos. Los directorios están afinados para dar una rápida respuesta a grandes volúmenes de búsquedas. [Microsoft. 2005]

Estos tienen la capacidad de replicar la información para incrementar la disponibilidad y la fiabilidad, al tiempo que reducen los tiempos de respuesta. Cuando la información de un directorio se replica, se pueden producir inconsistencias temporales entre las réplicas mientras esta se está sincronizando.

Hay muchas formas diferentes de proveer un servicio de directorio. Diferentes métodos permiten almacenar distintos tipos de información en el directorio, tener distintos requisitos sobre como la información ha de ser referenciada, consultada y actualizada, como es protegida de los accesos no autorizados, etc. Algunos servicios de directorio son locales, es decir, proveen el servicio a un contexto restringido. Otros servicios son globales y proveen servicio a un contexto mucho más amplio (como por ejemplo, Internet). Los servicios globales normalmente son distribuidos, esto significa que los datos están repartidos a lo largo de distintos equipos, los cuales cooperan para dar el servicio de directorio.

Típicamente, un servicio global define un espacio de nombres uniforme que da la misma visión de los datos, independientemente de donde se esté, en relación a los propios datos. El servicio DNS (Domain Name System) es un ejemplo de un sistema de directorio globalmente distribuido.

2.2.2.Qué es Active Directory

Active Directory es un integrado servicio de directorio distribuido que está incluido con Microsoft Windows Server 2003 y Microsoft Windows 2000 Server (ver figura 7). Integradas con Active Directory están muchas de las aplicaciones y servicios que previamente requerían un directorio separado y con distintos userid/password para ser manejados por cada aplicación o servicio. [Microsoft. 2005]

Con Active Directory, el administrador de la organización puede agregar un usuario al Active Directory y a través de una simple entrada habilitar el acceso remoto a la red, habilitar la misma cuenta de usuario para la mensajería Exchange, el mismo usuario para el acceso a la base de datos, administración de relaciones entre clientes, y otras aplicaciones.

Figura 7: Active Directory

Creando enlaces entre cuentas de usuarios, cuentas de correo, y aplicaciones, Active Directory simplifica la tarea de agregar, modificar, y eliminar cuentas de usuario. Cuando un usuario cambia su contraseña en Active Directory, él no necesita recordar diferentes contraseñas para sus otras aplicaciones. Esto es sólo un ejemplo de cómo Active Directory simplifica muchas de las tareas administrativas y procesos que, en el pasado, envolvían aplicaciones desiguales, servidores y servicios. [Microsoft. 2005]

2.2.3.Beneficios de Active Directory

Windows® 2003 Server y Active Directory ayuda a la pequeña y mediana organización con un fiable ambiente de trabajo para los usuarios finales, el cual ofrecen los más altos niveles de fiabilidad y rendimiento para que los usuarios puedan realizar su trabajo lo más eficientemente posible, así como también provee un ambiente seguro y manejable para hacer de las vidas de los encargados de TI mucho más fácil.

Ventajas de AD en diferentes áreas:

- Incrementa la productividad de los usuarios
- Reduce la carga de la Administración de TI
- Mejora la tolerancia a fallos para minimizar tiempos de caída
- Aumentando la seguridad para proveer tranquilidad al administrador

2.3.Cluster de computadores

Con el crecimiento explosivo de Internet y el incremento de la importancia de su rol en nuestras vidas diarias, el tráfico de Internet se ha incrementado dramáticamente, llegando a más del doble cada año. Sin embargo, como la demanda

y el tráfico incrementa, muchos sitios son desafiados a mantenerse activos, literalmente, y particularmente durante períodos de máxima actividad. Cualquier retraso o tiempo de desconexión puede ser desastroso, forzando a los clientes y beneficiarios a hacer algo. Esta demanda requiere del uso de hardware y software para construir servicios de red altamente disponibles y altamente escalables.

Cada día son más las compañías que mueven sus aplicaciones de misión crítica dentro de Internet. La demanda de un servicio de alta disponibilidad está creciendo; también la necesidad de servicios de red altamente disponibles y escalables. En general, un servicio con alta disponibilidad debe ser:

- E*scalable:* cuando la carga de trabajo del servicio incrementa, el sistema debe escalarse de acuerdo a los requerimientos.
- *Estar siempre disponible y habilitado*, ignorando fallas de parte del hardware o software.
- D*e costo efectivo:* el sistema completo debe ser económico de construir y expandir.
- Fácil de administrar.

Los cluster de servidores, interconectados en una red de alta velocidad, son una arquitectura viable y emergente para construir servicios de alto rendimiento y alta disponibilidad. Este tipo de arquitectura es más escalable, de menor costo efectivo, y más fiable que un sistema con un único procesador o un sistema multiprocesador. Sin embargo, éstos son los desafíos, incluyendo la transparencia y la eficiencia.

2.3.1.El Servidor Virtual Linux (LVS)

En 1998 se inició el *Linux Virtual Server* (LVS), proyecto que combina múltiples servidores físicos dentro de un servidor virtual, eliminando puntos únicos de fallos (SPOF, Single Point of Failure).

El Linux Virtual Server Project (LVS) permite un balanceo de carga de servicios de red tales como Web y servidores de correos utilizando Layer 4 Switching. Esto es extremadamente rápido y permite a tales servicios ser escalado a servicios de decenas, centenas o miles de conexiones simultáneas. [Zhang, 2005]

El Servidor Virtual Linux (LVS) habilita la multiplexación de conexiones entrantes de TCP/IP y datagramas UDP/IP hacia servidores reales. Este mecanismo de control de la conexión es a menudo referida como *Layer 4 Switching*. [Zhang, 2005]

Los paquetes son recibidos para un servicio virtual por un ***Linux Director***. El algoritmo de balanceo decide a cuál servidor real enviar el paquete. Una vez que esta decisión es hecha, los paquetes subsecuentes para la misma conexión serán enviados al mismo servidor real. Así, la integridad de la conexión es mantenida. Opcionalmente, un servicio puede ser marcado como persistente. Cuando esto es hecho todas las conexiones desde un host serán enviados al mismo servidor real. Esto es útil para aplicaciones tal como FTP donde los usuarios finales deberían consistentemente ser direccionados al mismo servidor real para asegurar la integridad de los datos de conexión. [Zhang, 2005]

El LVS por sí mismo se ejecuta en un servidor Linux, sin embargo, es permitido el balanceo de carga de conexiones desde usuarios finales ejecutando cualquier sistema operativo, hacia servidores reales ejecutando cualquier sistema operativo.

2.3.2.Servicios virtuales

En el Linux Director puede definirse un servicio virtual en: una dirección IP, puerto y protocolo; o una marca de firewall.

Direcciones IP, puerto y protocolo: un servidor virtual puede ser especificado por:

- Una dirección IP: la dirección IP que los usuarios finales usarán para

acceder al servicio.

- Un puerto: el puerto al cual los usuarios finales se conectarán.
- Un protocolo: cualquiera de UDP o TCP.

Marca de firewall: los paquetes pueden ser marcados con un valor de 32 bit sin signo utilizando ipchains/iptables. El servidor virtual Linux está habilitado para utilizar esta marca para asignar los paquetes destinados al servicio virtual y encaminarlos. Esto es particularmente útil si un largo número de servicios virtuales basados en IP contiguos son requeridos con los mismos servidores reales, o para agrupar persistentemente entre diferentes puertos. Por ejemplo, para asegurar que dado un usuario final es enviado el mismo servidor real para ambos protocolos, HTTP y HTTPS. [Robertson. 2004]

2.3.3.Algoritmos de balanceo

El servicio virtual es asignado a un algoritmo de balanceo para usar y asignar las conexiones entrantes a los servidores reales. Hay diferentes algoritmos de balanceo habilitados:

- *Round Robin (RR):* este algoritmo efectúa una conexión por cada equipo del cluster. Por ejemplo, si se tiene 3 máquinas en el cluster (A, B, C), la primera conexión sería a la máquina A, la segunda a la máquina B, la tercera a la máquina C y así sucesivamente.
- *Weighted Round Robin (WRR):* este algoritmo efectúa conexiones por prioridad utilizando el algoritmo anteriormente descrito (Round Robin). Por ejemplo, si se tiene 3 máquinas en al cluster (A, B, C) y sus prioridades son (4, 3, 2), las conexiones se efectuarían de la siguiente forma: AABACABBC (A como se ve, hace 4 conexiones, B hace 3 conexiones y C hace 2 conexiones, como se puede observar, la secuencia de conexiones sería esa secuencialmente).

- Least-Connection (LC): este algoritmo efectúa conexiones a la máquina del cluster con menos conexiones.
- *Weighted Least-Connection (WLC):* este algoritmo efectúa conexiones por prioridad de porcentaje, se le da un número a cada servidor del cluster con el cual se efectúan operaciones para asignar cuántas conexiones van a poder soportar cada equipo hasta pasar al siguiente de la lista de equipos del cluster. Es una combinación de las ventajas de *Least-Connection* y *Round Robin*. Es uno de los modos más utilizados.
- *Locality-Based Least-Connection (LBLC):* este algoritmo se queda con la dirección de destino (el cliente) y si el servidor en el que está efectuando operaciones se encuentra demasiado cargado, lo destina a otro servidor con menos carga. Se utiliza en los servidores de cluster con caché.
- *Locality-Based Least-Connection with Replication (LBLCR):* coge la dirección destino (del cliente) y mira a ver qué servidor está menos cargado, si todos los servidores están cargados espera hasta que pueda meterlo en algún servidor que deje alguna conexión libre y esté menos cargado. Se utiliza en los servidores de cluster con caché.
- *Destination-Hashing (DH):* asigna a la IP de destino un servidor del cluster específico haciendo un hash de la conexión.
- *Source-Hashing (SH):* asigna a la IP de origen un servidor del cluster específico haciendo un hash de la conexión.

Es necesario señalar que la elección del algoritmo de balanceo depende de cada instalación en particular, es importante estudiar previamente el balance del cluster, principalmente cuando recién se implementa, además, una de las ventajas del Linux Virtual Server es que el usuario puede modificar el algoritmo de balanceo en caliente, sin reiniciar el servicio.

2.3.4.Envío (forwarding) de paquetes

El Linux Virtual Server tiene tres caminos diferentes de enviar paquetes:

- *Network Address Translation (NAT):* es un método de manipulación del origen y/o puerto de destino y/o direcciones de un paquete de un mapa de redes. El uso más común de esto es el enmascaramiento IP que es usualmente utilizado para habilitar a redes privadas el acceso a direcciones de Internet. En el contexto de swtiching de capa 4, los paquetes son recibidos desde usuarios finales y el puerto de destino y la direción IP son cambiados para elegir al servidor real. Los paquetes de retorno pasan a través del dispositivo de switching de capa 4 y en cualquier tiempo del mapeo es pendiente de tal manera que el usuario final haya visto respuestas desde el origen esperado.
- *IP-IP encapsulation (Tunnelling):* permite que los paquetes direccionados a una dirección IP sean redireccionados hacia otra dirección IP, posiblemente en una red diferente. En este contexto de swtiching de capa 4 el comportamiento es muy similar al direct routing, excepto que cuando los paquetes son enviados hay que encapsularlos en un paquete IP, en vez de sólo estar manipulando el cuadro ethernet. La principal ventaja del uso del tunneling es que los servidores reales pueden ser de diferentes redes.
- *Direct Routing:* los paquetes desde los usuarios finales son enviados directamente hacia el servidor real. El paquete IP no es modificado, por lo que los servidores reales deben ser configurados para aceptar el tráfico para la dirección IP de servidor virtual. Esto puede ser hecho con una interfaz falsa, o filtrando los paquetes para redireccionar las direcciones de tráfico hacia las direcciones IP de los servidores virtuales hacia un puerto local. El servidor real puede enviar respuestas directamente de vuelta al usuario final. Esto es, si es utilizado el host basado en switch capa 4, que puede no estar en el path de retorno.

2.3.5.Servidores Reales

El servidor real pertenece a un servidor virtual y es definido por una dirección

IP, un puerto y, opcionalmente, un peso. Si es pesada se usan los algoritmos de Weighted Least-Connection o Weighted Round Robin. En general, el puerto del servidor real debe ser el mismo que el del servicio virtual. Si se utiliza marcas de firewall en los servicios virtuales, entonces el puerto de conexiones de entrada que es recibida será el puerto de la conexión que se envía al servidor real, esta es la especificación del puerto para que no se use en el servidor real. Los latidos de los servidores reales pueden ser monitoreados utilizando *ldirectord*.

2.3.6.Heartbeat

Heartbeat es un proyecto Open Source que implementa un protocolo Heartbeat. Esto es, mensajes son enviados durante intervalos regulares entre dos máquinas y si un mensaje no es recibido desde una máquina en particular, entonces se asume que la máquina tiene una falla y alguna forma de acción evasiva es tomada. Heartbeat puede enviar mensajes mediante interfaces seriales y/o interfaces de red. [Horman, 2005]

Cuando Heartbeat es configurado, es seleccionado un nodo master. Cuando Heartbeat inicia, este nodo establece una interfaz para una dirección IP virtual, la cual es accedida por los usuarios finales externos. Si este nodo falla, entonces otro nodo en el cluster Heartbeat levantará una interfaz para esa dirección IP virtual y usará “ARP no pedido” para asegurar que todo el tráfico con destino a esa dirección IP virtual sea recibido por esta máquina. Este método de failover es llamado IP Address Takeover. Dependiendo de las directivas de configuración, una vez que el nodo master vuelva a estar habilitado, los recursos son devueltos al nodo master.

Cada dirección IP virtual es asociada a un recurso, un programa que Heartbeat iniciará al iniciar y finalizará al apagar. Heartbeat permite arbitrariedad de los recursos a ser usados.

2.3.6.1.Fallo del medio de comunicación

Si Heartbeat pierde la comunicación con el otro nodo entonces lo marcara como un nodo inasequible. Si la pérdida de comunicación ocurre porque ha fallado el medio de comunicación, entonces ambos nodos marcarán al otro como un nodo inasequible y tomarán los recursos. [Horman, 2005]

En el caso de que la dirección IP tomada, esto puede resultar con dos máquinas teniendo la misma dirección IP. Esto puede no ser crítico cuando el nodo no se puede comunicar con la red de cualquier forma. Una vez que la comunicación de la red es restaurada, entonces Heartbeat debería resolver esto y sólo un nodo tomará la dirección IP virtual. Sin embargo, para algunos recursos -por ejemplo un disco compartido- esto puede ser un problema crítico. [Zhang, 2005]

El mejor camino de evitar esto es asegurarse que los fallos del medio de comunicación no ocurrirán cuando se tienen múltiples enlaces entre los nodos. Una o más tarjetas de red y uno o más enlaces seriales son recomendados. Es generalmente conveniente configurar Heartbeat para comunicarse sobre todos los enlaces habilitados.

2.3.6.2.Absorción (takeover) de la dirección IP

Si una máquina, o servicio ejecutándose en una máquina, se convierte en inasequible, es a menudo útil sustituirlo con otra máquina. La máquina es a menudo referida como un host pasivo. En el caso más simple, la dirección IP absorbida involucra dos máquinas, cada una con una dirección IP, que es utilizada para acceso administrativo. Adicionalmente, hay una dirección IP virtual que es accedida por los usuarios finales. La dirección IP virtual será asignada a uno de los servidores, el master.

La absorción de la dirección IP empieza cuando el host pasivo levanta una interfaz para la dirección IP virtual. Esto es mas conveniente hacerlo utilizando un IP

alias, esto es, configurando una segunda interfaz lógica en una interfaz física existente. Una vez que la interfaz está habilitada, el host pasivo está habilitado para aceptar el tráfico, y responder peticiones ARP, mediante la dirección IP virtual. Sin embargo, esto no asegura que todo el tráfico de la dirección IP virtual será recibido por el host pasivo.

Aunque el nodo master puede ser inaccesible, puede mantenerse capaz de responder peticiones ARP para la dirección de hardware (MAC) de la dirección IP virtual. Si esto ocurre, entonces cada vez que un host en la LAN envíe una petición ARP será una condición de carrera, y potencialmente los paquetes serán enviados al master que ha sido determinado a tener falla en algún caso. Adicionalmente, aún si el host master no emite respuestas ARP, el tráfico continuará siendo enviado a la interfaz del host master. Esto continuará hasta que las entradas de caché de ARP de los otros hosts y routers en la red expiren.

Para acelerar el fail-over y asegurar que todo el tráfico irá al host pasivo, una técnica conocida como ***ARP no pedido*** será utilizada. Usualmente el ARP funciona como sigue. El host A envía una petición ARP a la dirección de hardware de una dirección IP de un host B. El host B mira las peticiones y envía una respuesta ARP conteniendo la dirección de hardware de la interfaz con la dirección IP en cuestión. El host A entonces guarda la dirección de hardware en su caché ARP de tal manera que no tenga que hacer una petición ARP y esperar una respuesta cada vez que quiera enviar un paquete. Las entradas de la caché ARP típicamente expiran después de alrededor 2 minutos. Un ARP no pedido es una respuesta ARP cuando no hubo una petición ARP. Si la respuesta ARP es enviada a la dirección de hardware de broadcast, entonces todos los hosts de la LAN recibirán la respuesta ARP para la dirección IP en cuestión antes que expire, de esta manera no serán enviadas peticiones ARP, por lo tanto no hay oportunidad de colocar una respuesta ARP desde el nodo que ha fallado que está fuera. [Horman, 2005]

Al abandonar una dirección obtenida a través de una absorción de dirección IP

la interfaz de la dirección virtual debería bajarse. Más allá, para asegurar una rápida transición, los ARP no pedido deberían ser emitidos con la dirección de hardware de la interfaz del host master con la dirección virtual. Dependiendo del servicio, puede ser mejor revertir los roles del host pasivo y el master una vez que el nodo master halla fallado y vuelve a estar en línea, en vez de cancelar el failover. Para hacer esto efectivamente los host necesitarán negociar la propiedad de la dirección IP virtual, idealmente utilizando el protocolo Heartbeat.

Los ARP no pedidos pueden ser utilizados maliciosamente para absorber la dirección IP de una máquina. Por esto, algunos routers y switches ignoran, o pueden ser configurados para ignorar ARP no pedidos. En una red dada, esto puede o no puede ser una emisión, pero para que la absorción de dirección IP sea satisfactoria, el equipamiento debe ser configurado para aceptar ARP no pedidos o limpieza de la caché ARP en caso necesario. Aparte de esto no hay problemas conocidos cuando se utiliza ARP no pedido y, por lo tanto, la absorción de direcciones IP en ambas redes ethernet switcheadas y no switcheadas será posible. [Horman, 2005]

2.3.7.Dinámica HA

Se considera dinámica HA (High Availability, o Alta Disponibilidad) a todas las reconfiguraciones del cluster que garanticen la máxima disponibilidad del servicio de datos. Esta dinámica está orientada a los nodos integrantes del cluster y la forma en la cual el cluster responde. Se denomina de la siguiente manera: [Zhang y Zhang, 2005]

2.3.7.1.Failover

Es un término genérico que se usa cuando un nodo debe asumir la responsabilidad de otro nodo, importar sus recursos y levantar el servicio de datos. Se ha de entender que una situación de failover es una situación excepcional, para la cual la alta disponibilidad ha sido concebida, el fallo de un nodo. Si sólo queda un nodo en el cluster, tras los fallos de los demás, se estará en un SPOF (Punto único de fallo)

hasta que el administrador de sistema verifique y restaure el cluster. También se ha de entender que el servicio de datos sigue levantado, que es el objetivo de la alta disponibilidad.

2.3.7.2.Takeover

Es un failover automático que se produce cuando un nodo nota el fallo en el servicio de datos. Para ello debe haber cierta monitorización con respecto al servicio de datos. El nodo que se declara fallido es forzado a ceder el servicio y recursos, o simplemente eliminado.

2.3.7.3.Swtichover o Giveaway

Es un failover manual, consiste en ceder los recursos de un servicio de datos y este mismo a otro nodo del cluster, mientras se realizan ciertas tareas administrativas. A este procedimiento se le denomina “Node outage”.

2.3.7.4.Splitbrain

Para la gestión de un cluster de alta disponibilidad es necesario un mecanismo de comunicación y verificación entre nodos integrantes. Por este mecanismo, cada nodo debe gestionar los recursos que corresponden a cada uno, según el estado del cluster; a su vez, cada nodo debe hacer chequeos o latidos (beats) a sus compañeros. Un splitbrain (división de cerebros) es un caso especial de failover, en el cual falla el mecanismo de comunicación y gestión del cluster de dos nodos. Es una situación en la cual cada nodo cree que es el único activo, y como no puede saber el estado de su nodo compañero, tomará acciones en consecuencia, forzando un takeover. Esta situación es peligrosa, los dos nodos intentarán apropiarse de todos los recursos, incluyendo el servicio de datos. El peligro aumenta sobre todo cuando se tienen recursos delicados, como recursos de almacenamiento, ya que cada nodo podría tomar y escribir por su cuenta y quebrar la integridad de datos.

Para evitar este problemas, cada nodo debe actuar de una forma prudente, y utilizar los recursos compartidos como señal que está vivo. La forma de proceder de cada nodo es similar, ya que cada uno cree que su compañero ha desaparecido.

Otra forma de evitar esta situación, un poco más violenta, es que un nodo elimine a su compañero; el primero que apague a su compañero se queda con todos los recursos. Es un mecanismo muy brusco, pero muy eficaz. Para este caso existen tarjetas especializadas para esta tarea.

2.3.8.Punto simple de fallo (SPOF - Single Point of Failure)

La regla básica para hacer un sistema de alta disponibilidad es la replicación de elementos. Con SPOF se quiere hacer referencia a cualquier elemento no replicado y que puede estar sujeto a fallos; afectando con ello al servicio. Se debe evitar SPOF en cualquier subsistema del sistema, ya que en caso de fallo puede peligrar el servicio de datos por culpa de un subsistema. [Horman, 2005]

En el caso de alta disponibilidad (con hardware y software adecuado) se puede reemplazar un adaptador de red o un servidor automáticamente. No sólo los servidores han de ser redundantes sino los elementos que facilitan el servicio, como routers, bridges o la propia red local.

Es conveniente olvidar la redundancia a cierto nivel, pues habrá un nivel en que la alta disponibilidad se hace extremadamente cara.

2.3.9.Ldirectord

Ldirectord monitorea los latidos de los servidores reales periódicamente solicitando una URL conocida y chequeando que la respuesta contenga un string (cadena de caracteres) esperado. Este es llamado un chequeo de “negociación”.

Ldirectord tiene soporte para servidores HTTP, HTTPS, FTP, IMAP, POP, SMTP, LDAP y NNTP. Añadir otros soportes a otros protocolos es usualmente muy simple.

Si un servidor real falla, entonces el servidor queda inactivo y será reinsertado una vez que vuelva a estar en línea.

2.3.10.Topologías

Existen variadas topologías de cluster, cada una de ellas destinada a una solución en particular que se deben estudiar para poder elegir la mejor topología que se adapte al resultado que se desea obtener. Las topologías que se verán a continuación son: [Horman, 2005]

- Estándares
    - Alta disponibilidad
    - Balanceo de carga
    - Alta disponibilidad y balanceo de carga
- Avanzadas
    - Alta capacidad, alta disponibilidad y balanceo de carga
    - Alta disponibilidad y balanceo de carga simplificada

Usando marcas de firewall

Las marcas de firewall proveen una poderoso mecanismo para agrupar los servicios juntos. Ellos hacen que el uso de poderosos frameworks de filtrado de paquetes para hacer juego con el tráfico destinado a una servicio virtual y marcar estos paquetes internamente con una marca de firewall. Esta marca de firewall es utilizada por LVS para identificar los paquetes que deben ser balanceados y enviados a los servidores reales.

2.3.10.1.Estándares

2.3.10.1.1.Alta disponibilidad

Esta topología provee un servicio altamente habilitado utilizando los mínimos requerimientos de hardware.

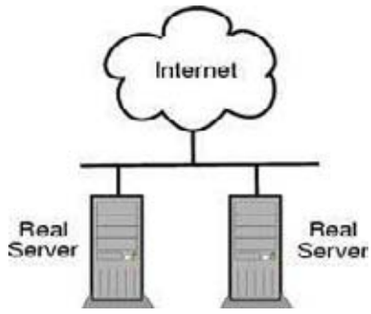


Figura 8: Alta disponibilidad de servidores

Componentes: Hay dos servidores reales monitoreándose uno a otro usando Heartbeat y en el evento del fallo del servidor activo el host pasivo asumirá la dirección virtual y el servicio será mantenido.

2.3.10.1.2.Balanceo de carga

Esta topología se enfoca en proveer un servicio balanceador de carga con los mínimos requerimientos de hardware. Los servidores reales pueden ser añadidos como servicios adicionales cuando son requeridos.

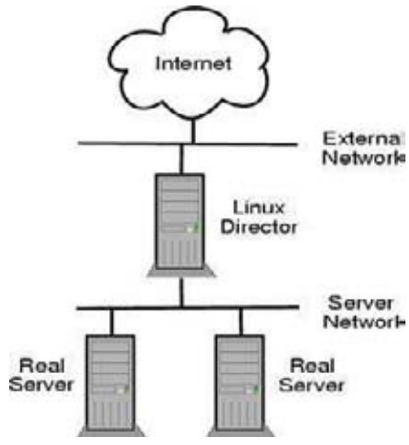


Figura 9: Balanceo de carga de servidores

Linux Director: actúa como el gateway por defecto de la red de servidores

reales y balancea la carga del tráfico utilizando LVS. El Linux Director puede ser típicamente adecuado con dos interfaces de red (NICs). Opcionalmente, el Linux director puede tener habilitado el filtrado de paquetes para controlar el tráfico que dejará entrar y salir de la red de servidores reales.

Cuando un Linux director recibe una conexión desde un usuario final, toma la decisión hacia cuál servidor real enviará la conexión. Todos los paquetes de por vida de esta conexión serán enviados al mismo servidor real de tal manera que se mantenga la integridad de la conexión entre el usuario final y el servidor real.

Ldirectord monitorea los latidos de los servidores reales a través de una petición de una página conocida periódicamente y chequeando que la respuesta contenga una cadena esperada. Si un servidor real falla, entonces el servidor es sacado de la granja de servidores reales y será reinsertado una vez que vuelva a estar en línea.

Servidores Reales: los servidores reales pueden ejecutar una gran variedad de servicios. Adicionalmente, se pueden agregar servidores reales a la red para obtener capacidad extra cuando sea requerida.

2.3.10.1.3.Alta disponibilidad y balanceo de carga

Esta topología provee el servicio de alta disponibilidad y balanceo de carga. Se requiere un mínimo de 4 nodos para esta topología. Los servidores reales pueden ser añadidos a la red cuando sea requerida capacidad adicional.

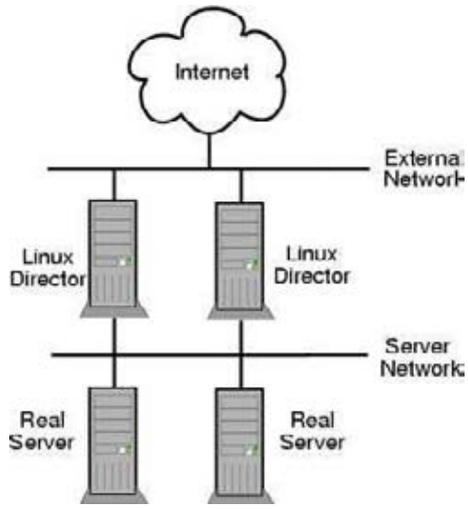


Figura 10: Alta disponibilidad y balanceo de carga

Linux Directors: en cualquier tiempo dado un Linux Director está activo. El director activo acepta el tráfico para una dirección IP virtual y balancea el tráfico utilizando LVS. La dirección IP virtual está típicamente registrada a los usuarios finales a través del DNS. Cada Linux director monitorea al otro utilizando Heartbeat, y en el evento de falla del Linux director activo, el Linux director pasivo asumirá la dirección IP virtual y mantendrá el acceso al servicio.

El Linux director activo actúa como gateway de la red y puede ser típicamente adecuado con dos interfaces de red (NICs). Opcionalmente, los Linux directores pueden tener habilitado el filtrado de paquetes para controlar el tráfico que deja entrar y salir de la red de servidores reales.

Cuando un Linux director recibe una conexión desde un usuario final, toma la decisión hacia cuál servidor real enviará la conexión. Todos los paquetes de por vida de esta conexión serán enviados al mismo servidor real de tal manera que se mantenga la integridad de la conexión entre el usuario final y el servidor real.

Ldirectord monitorea los latidos de los servidores reales a través de una petición de una página conocida periódicamente y chequeando que la respuesta contenga una cadena esperada. Si un servidor real falla, entonces el servidor es

sacado de la granja de servidores reales y será reinsertado una vez que vuelva a estar en línea.

Servidores Reales: los servidores reales pueden ejecutar una gran variedad de servicios. Adicionalmente, se pueden agregar servidores reales a la red para obtener capacidad extra cuando sea requerida.

2.3.10.2.Avanzadas

2.3.10.2.1.Alta capacidad, altamente disponible y balanceo de carga

Esta topología habilita el máximo rendimiento a través de la red porque el tráfico de retorno tiene que viajar a través de un Linux director. Esta topología se construye en una infraestructura de red existente que permita políticas de ruteo interno para ordenar el tráfico de salida.

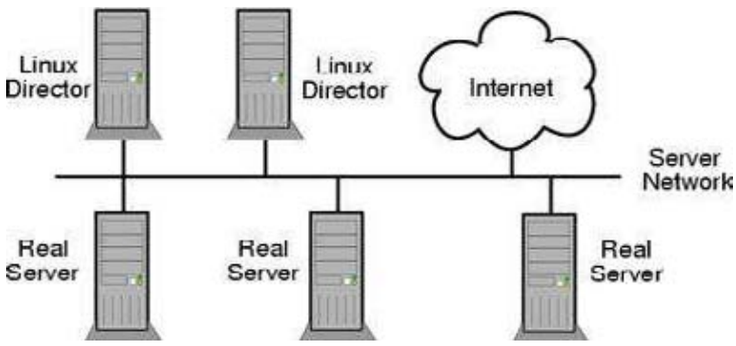


Figura 11: Alta capacidad, altamente disponible y balanceo de carga

Linux Directors: en cualquier tiempo dado un Linux Director está activo, mientras el otro es una máquina pasiva. El Linux director activo acepta el tráfico para una dirección virtual y balancea el tráfico utilizando LVS. La IP virtual está típicamente registrada a los usuarios finales a través del DNS. Cada Linux director monitorea al otro utilizando Heartbeat, y en el evento de falla del Linux director activo, el Linux director pasivo asumirá la IP virtual y mantendrá el acceso al servicio.

Cuando un Linux director recibe una conexión desde un usuario final, toma la decisión hacia cuál servidor real enviará la conexión. Todos los paquetes de por vida de esta conexión serán enviados al mismo servidor real de tal manera que se mantenga la integridad de la conexión entre el usuario final y el servidor real.

Ldirectord monitorea los latidos de los servidores reales a través de una petición de una página conocida periódicamente y chequeando que la respuesta contenga una cadena esperada. Si un servidor real falla, entonces el servidor es sacado de la granja de servidores reales y será reinsertado una vez que vuelva a estar en línea.

Como los Linux Directors no son gateway de los servidores reales, el tráfico de retorno no tiene que viajar a través del Linux director si se utiliza direct routing o tunneling. Esto habilita el mejor rendimiento como los Linux directores no tienen que manejar los paquetes de retorno. Esto puede ser posible enviando el tráfico de retorno a través de diferentes routers y/o uplinks como conexiones externas permita.

Servidores Reales: los servidores reales pueden ejecutar una gran variedad de servicios. Se pueden agregar servidores reales a la red para obtener capacidad extra cuando sea requerida.

2.3.10.2.2.Alta disponibilidad y balanceo de carga simplificada

Esta topología combina las funciones de Linux director y servidor real dentro de las mismas máquinas, así la alta disponibilidad y el balanceo de carga puede ser mantenido con sólo dos nodos.

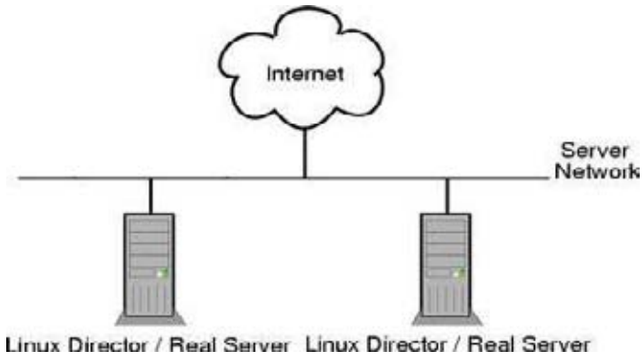


Figura 12: Alta disponibilidad y balanceo de carga simplificada

Linux Director / Real Server: en cualquier tiempo dado uno de los hosts está activo como Linux Director, mientras que el otro se encuentra en stand-by. EL Linux director activo acepta el tráfico a través de la dirección IP virtual. Esta es la dirección IP virtual que está registrada a los usuarios finales a través del DNS. Las conexiones son balanceadas hacia ambos, el host local y el otro nodo utilizando LVS. Cada Linux director monitorea al otro utilizando Heartbeat, y si un evento de falla del Linux director activo, el Linux director pasivo asumirá la IP virtual y mantendrá el acceso al servicio.

Cuando un Linux director recibe una conexión desde un usuario final, toma la decisión hacia cuál servidor real (él mismo o el otro nodo) enviará la conexión. Todos los paquetes de por vida de esta conexión serán enviados al mismo servidor real de tal manera que se mantenga la integridad de la conexión entre el usuario final y el servidor real.

Ldirectord monitorea los latidos de los servidores reales a través de una petición de una página conocida periódicamente y chequeando que la respuesta contenga una cadena esperada. Si un servidor real falla, entonces el servidor es sacado de la granja de servidores reales y será reinsertado una vez que vuelva a estar en línea.

El retorno de las conexiones está directamente dentro de la red de área local, o a través de un gateway. Cuando se utiliza direct routing en la configuración, los paquetes de retorno no necesitan volver a pasar a través del balanceador de carga.

Los servidores pueden ejecutar una gran variedad de servicios. Adicionalmente, se puede agregar servidores reales a la red (pero no actuando como Linux Director) para incrementar la capacidad cuando sea requerida.

2.4.Samba

Samba es una suite de Software Open Source/Free que provee un conjunto de servicios de archivo e impresión para clientes que utilizan el protocolo SMB (Server Message Block) y/o CIFS (Common Internet File System). Samba es disponible libremente - a diferencia de otras implementaciones SMB/CIFS - y permite la interoperabilidad entre servidores Linux/Unix y clientes basados en Windows. [Samba Team. 2004]

Samba es el software que puede ejecutarse en plataformas distintas a Microsoft Windows, por ejemplo, Unix, Linux, IBM System 390, OpenVMS, y otros sistemas operativos. Cuando está correctamente configurado, este permite que un host interactúe con un cliente Microsoft Windows o como servidor si éste es un servidor de archivos e impresión.

2.4.1.Funcionalidades

Samba consiste en dos programas: smbd y nmbd. Su trabajo es implementar los cuatro servicios básicos de CIFS:

- Servicios de archivos e impresión.
- Autenticación y autorización: gestiona los códigos de acceso de una red

Windows, es decir, se puede utilizan Samba pata autenticar a los usuarios de una red.

- Resolución de nombres: puede funcionar como servidor WINS (primario o secundario) para la resolución de nombres.
- Anuncio de servicios (browsing): puede acceder a recursos compartidos en máquinas con Windows 9/NT desde estaciones Linux/UNIX.

Los servicios de archivo e impresión son, por supuesto, la base de la suite CIFS. Estos están provistos por smbd, el demonio SMBD. Smbd también maneja la autenticación y autorización “share mode” y “user mode”. Esto es, que uno puede proteger archivos compartidos y servicios de impresión requiriendo passwords.

2.4.2.Características propias de Samba

Las características propias de Samba son: [Gómez, 2000]

- Incorporación de un cliente tipo FTP para poder acceder a los recursos compartidos (tanto a los archivos como impresoras).
- Posibilidad de realizar copias de seguridad mediante la utilidad “tar”.
- Herramienta de configuración gráfica vía Web, llamada Swat.
- Posibilidad de acceder a los recursos compartidos por Samba mediante un navegador Web.

2.4.3.Autenticación de usuarios con Samba

Samba ofrece dos métodos para autenticar usuarios Active Directory (AD): [Samba Team. 2005]

- *Usando cuentas locales:* utilizando este método, todos los usuarios tienen requerido tener una cuenta local en el archivo /etc/passwd. Cuando un cliente intenta una conexión, el username y el password son pasados al

Controlador de Dominios (DC) para la autentificación. Toda la información de usuarios y grupos (con la excepción del password) está almacenada en archivos estándares de Linux.

- *Usando Winbind:* Winbind es una extensión de Samba el cual hace posible ejecutar el servidor Samba sin mantener separadas las cuentas Unix para los usuarios AD. Winbind archiva esto creando una cuenta Unix necesaria en el momento. Estas cuentas son asignadas con un único UID (o GID para grupos). Enlaza la información de estas cuentas con su equivalente Active Directory y es transparentemente mantenido por el demonio Winbind.

2.4.4.Configuración de Samba

La configuración de samba está dividida mediantes secciones, identificadas por los caracteres “[“ y “]”. La principal sección, en la cual se definen parámetros globales es la que se indica a continuación:

```
[global]
workgroup = DOMINIO
netbios name = samba server
server string = servidor samba
security = ads
realm = DOMINIO.COM
password server = 172.26.192.8
encrypt password = yes

local master = no
domain master = no
prefered master = no
wins server = 172.26.192.4
dns proxy = no
```

En esta configuración de ejemplo se especifica el grupo de trabajo (workgroup) de la red; a continuación un nombre identificatorio del servidor (netbios name y server string), para posteriormente indicarle a Samba que se utilizará un servidor Active Directory, el cual posee un dominio (real), un servidor de password (password server) y que las passwords viajan por la red en forma encriptada (encrypt password).

Finalmente, se especifica que este servidor Samba no será un controlador de doninio, y que utilice un servidor wins ya existente (wins server).

2.4.5.Winbind

Winbind es un componente de la suite de programas Samba que permite la integración de Sistemas Operativos UNIX y Microsoft Windows a través de un logon unificado. Winbind utiliza la implementación UNIX de las llamadas RPC de Microsoft. Pluaggable Authentication Modules (PAM) y el Name Service Swtich (NSS) que permiten a los usuarios de un dominio Active Directory aparecer y operar como usuarios UNIX en una máquina UNIX. [Samba Team. 2005]

Winbind puede ser también usado en conjunto con un PDC Samba, así se elimina la necesidad de un servidor Microsoft en la red. Esto puede proveer un mecanismo simple para implementar una única fuente de login en toda una red Unix.

Winbind es el servicio para resolver información de usuarios y grupos desde servidores Windows NT. El paquete provee el demonio Winbind, el cual provee un servicio de Name Service Switch que está presente en las mas modernas librerías de C (como glibc). El servicio también provee servicios de autenticación vía módulos de PAM. [Samba Team. 2005]

2.4.5.1.Configurando Samba para utilizar Winbind

Para integrar Winbind a la configuración de Samba, tan sólo es necesario agregar unas pocas líneas al archivo de configuración inicial smb.conf.

Para agregar soporte a Winbind, hay que agregar la siguiente línea a la sección [global]:

winbind separator = +

Este parámetro de winbind es utilizado para separar el dominio y el username con la letra “+”, por ejemplo: DOMINIO+deugenin. Si no se utiliza esta línea, el separador por defecto es “\”, tal como es normalmente utilizado en máquinas Windows para separar el DOMINIO del username. El problema para Linux al utilizar winbind con el separador “\” es que este carácter es interpretado como un carácter de escape. En ciertas situaciones, se tiene que definir más de un “\” como separador (por ejemplo: DOMINIO\\usuario), o el comando fallará. Esto puede ser un poco confuso para usuarios inexperimentados, por lo cual se recomienda la utilización del carácter “+”.

winbind uid = 10000-20000

Este parámetro de winbind especifica el rango de UID (User ID) que son asignados por el demonio winbind. Este rango de UIDs no debería existir en las cuentas locales que posea el servidor Linux.

winbind gid = 10000-20000

Este parámetro de winbind especifica el rango de GID (Group ID) que son asignados por el demonio winbind. Este rango de GIDs no debería existir en las cuentas locales que posea el servidor Linux.

winbind enum users = yes
winbind enum groups = yes

Estos parámetros de winbind permiten la enumeración de usuarios y grupos winbind Sólo en raras situaciones se debería setear estos parámetros en “no”,

winbind use default domain = yes

Este parámetros de winbind elimina la necesidad de utilizar el dominio (domain name) con el nombre usuario/grupo. El nombre de dominio más el separador automáticamente serán antepuestos al nombre de usuario (user name).

Estos son los únicos parámetros que se deben agregar al archivo smb.conf para hacer que Samba utilice Winbind.

2.4.5.2.Test de Winbind

Antes de realizar el testeo de Samba y Winbind es necesario ingresar el servidor dentro del dominio Active Directory, escribiendo el siguiente comando:

net ads join admin%DOMINIO

Luego se debe ingresar la contraseña del administrador de Active Directory y el servidor quedará inmediatamente agregado al dominio.

Una vez que Samba y Winbind han sido iniciados -y que además el servidor haya sido agregado al dominio Active Directory-, es hora de testear que todo ha quedado correctamente. Existen diversos comandos que se pueden utilizar para asegurarse que todo está operando normalmente.

wbinfo -u

Esto debería devolver la lista de los usuarios en el dominio.

wbinfo -g

Esto debería devolver la lista de los grupos en el dominio.

2.4.5.3.Configuración final de Samba con Winbind

Finalmente, la configuración de Samba quedaría de la siguiente manera:

```
[global]
workgroup = DOMINIO
netbios name = samba server
server string = servidor samba
security = ads
realm = DOMINIO.COM
password server = 172.26.192.8
encrypt password = yes

local master = no
domain master = no
prefered master = no
wins server = 172.26.192.4
dns proxy = no

winbind separator = +
winbind uid = 10000-20000
winbind gid = 10000-20000
winbind enum users = yes
winbind enum groups = yes
winbind use default domain = yes
```

2.5.Kerberos

Kerberos es un protocolo de autentificación que posee la característica de utilizar un tercer servidor en el que todos los integrantes de la red, usuarios y máquinas, confían y con el que todos comparten un secreto, que generalmente será una clave o algo derivado de ella. [Lucas, 2000]

Kerberos fue desarrollado originalmente para el proyecto Athena del MIT, para cuyo desarrollo se estudiaron primero los protocolos ya existentes sobre autentificación, especialmente aquellos en los que intervenía un servidor de confianza, y finalmente se aprovechó el modelo creado por Neddham y Schroeder, que precisamente lleva su nombre.[Lucas, 2000]

Con algunas ligeras modificaciones nació Kerberos, a mediados de los ochenta. Al principio, dado el carácter del proyecto Athena, las primeras versiones de

Kerberos permanecieron en secreto. Fue al llegar a la versión 4 del protocolo cuando se decidió liberar el código y, en definitiva, el protocolo, para que todo el mundo pudiera disfrutar de él. No obstante, no se contó con una limitación, que Kerberos utiliza una encriptación considerada como fuerte por las leyes de exportación de Estados Unidos, cosa que frenó gravemente su expansión. [Lucas, 2000]

Los integrantes del proyecto Athena del MIT deseaban que Kerberos pudiese ser exportado desde los Estados Unidos. Para ello trataron de conseguir los permisos necesarios pero, visto que no lograban nada, decidieron eliminar la parte criptográfica de Kerberos, creando así una nueva versión. Esta versión se llama Bones y es la única que por el momento se puede conseguir de forma legal fuera de Estados Unidos. Mientras por un lado continuaba la lucha por exportar Kerberos fuera de Estados Unidos, el protocolo iba evolucionando por su cuenta hasta llegar a la versión 5, que es la más reciente. [Lucas, 2000]

2.5.1.Funcionamiento de Kerberos

Como ya se mencionó anteriormente, Kerberos se basa en el uso de un tercer participante en el proceso de autentificación. Este tercer participante es el que se encarga de gestionar las claves de los usuarios y de los servidores.

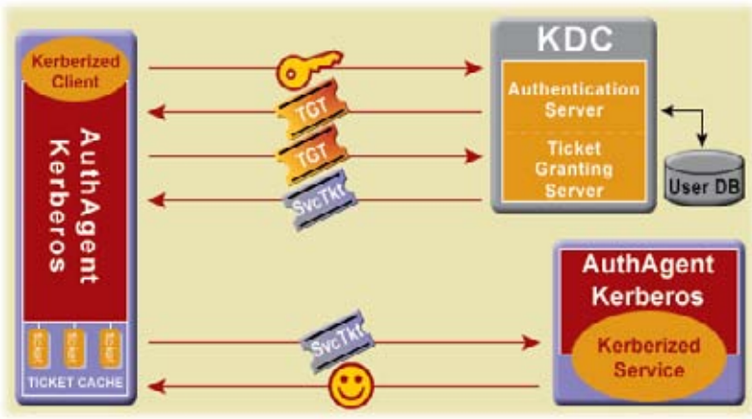


Figura 13: Funcionamiento general de Kerberos

Kerberos introduce dos credenciales que serán utilizadas durante el proceso de autentificación. La primera, en la que se basa todo el proceso, es el *ticket*.

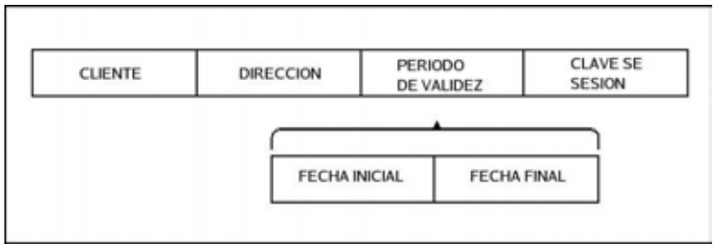


Figura 14: Formato de un ticket de Kerberos

Un *ticket* tiene la estructura que aparece en la figura 14, donde se puede observar cómo se incluye información sobre el usuario, información temporal para garantizar la validez del ticket y una clave de sesión, que será la utilizada para realizar el acceso al servidor.

Además del ticket, se tienen los *autentificadores*, que son como los mostrados en la figura 15. Estos autentificadores contienen la identificación del cliente, el instante en que se solicita el servicio y, además, puede incluir de forma opcional una clave para cifrar la sesión.

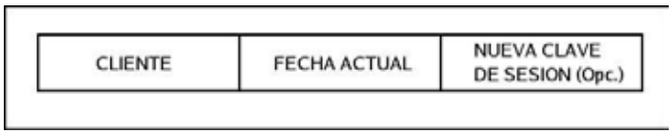


Figura 15: Formato de un autentificador de Kerberos

Con estos elementos, Kerberos define un protocolo muy sencillo de entender, por ser prácticamente cotidiano.

Cuando el cliente quiere acceder a un servidor concreto solicita a Kerberos un ticket. En Kerberos 5 este ticket no es para acceder directamente al servidor, sino para acceder a otro servicio llamado Servidor de Concesión de Tickets (TGS). Este ticket que devuelve Kerberos sólo sirve para que cada vez que el cliente quiera acceder al servidor, durante el periodo de validez de este ticket, se pueda solicitar otro

al TGS sin tener que comprometer la clave del cliente. Por eso, el ticket proporcionado por Kerberos también posee un nombre especial: Ticket para la Concesión de Tickets (TGT).

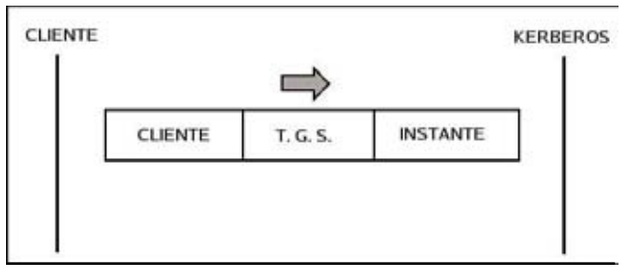


Figura 16: Primer paso del protocolo Kerberos

El cliente guarda el TGT durante un período de validez de éste y cada vez que quiere acceder a un servicio, lo usa para solicitar al TGS el ticket que permita acceder al servidor.

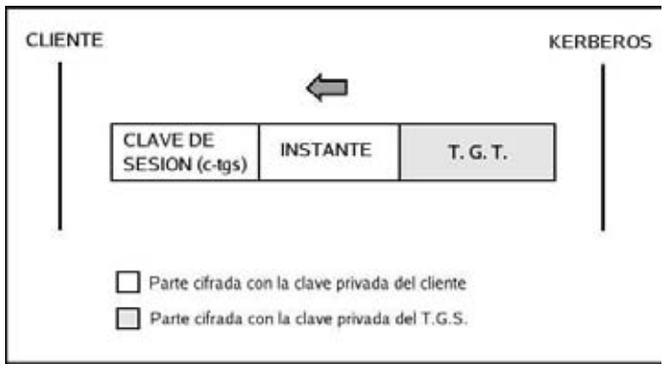


Figura 17: Segundo paso del protocolo Kerberos

Como se puede observar, el factor tiempo es muy importante al utilizar Kerberos, por lo que se exige una perfecta sincronización entre todas las máquinas de la red. Esta característica introduce un elemento de seguridad importante, que es la validez temporal, pero a la vez crea otros problemas.

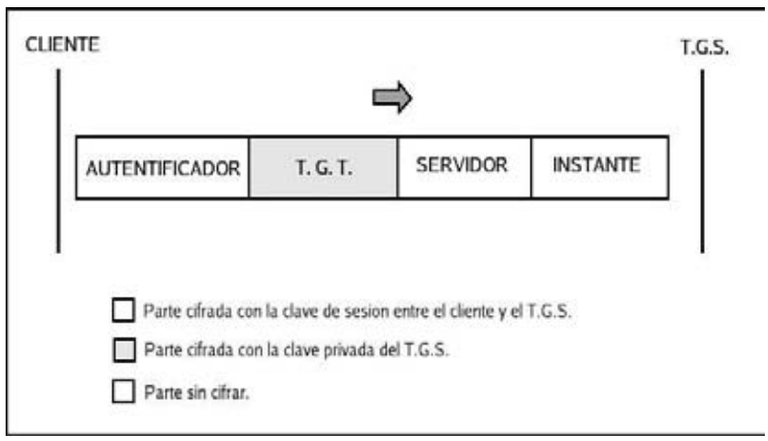


Figura 18: Tercer paso del protocolo Kerberos

En las figuras 16 a la 20 se ilustran los diferentes pasos del protocolo Kerberos, mientras que en la 21 se ve el diagrama de funcionamiento de este protocolo.

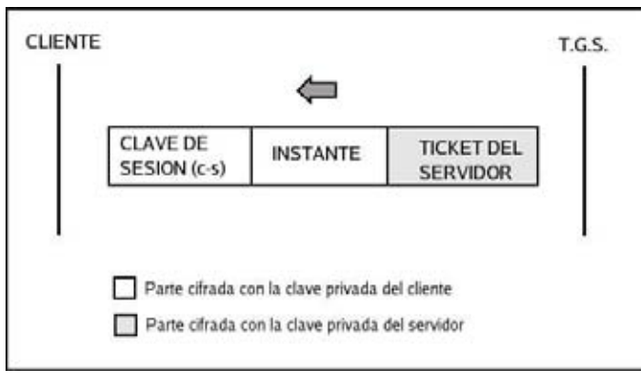


Figura 19: Cuarto paso del protocolo Kerberos

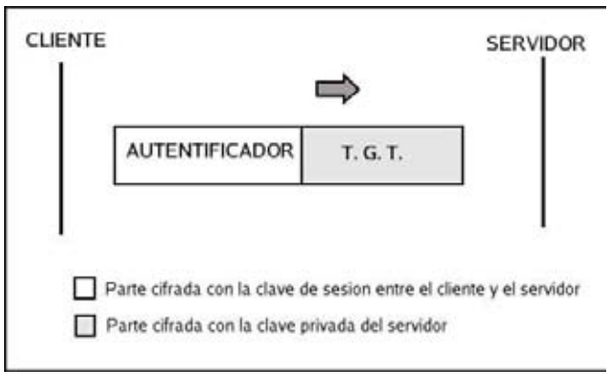


Figura 20: Quinto paso del protocolo Kerberos

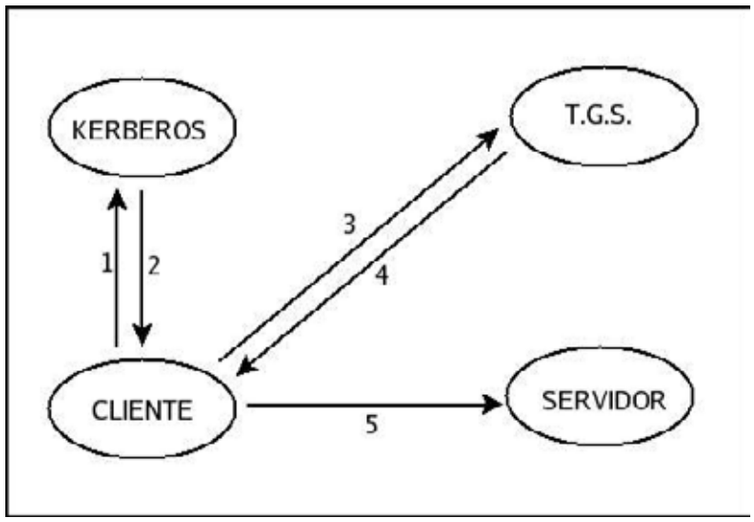


Figura 21: Diagrama de funcionamiento de Kerberos

Resumen del funcionamiento de Kerberos:

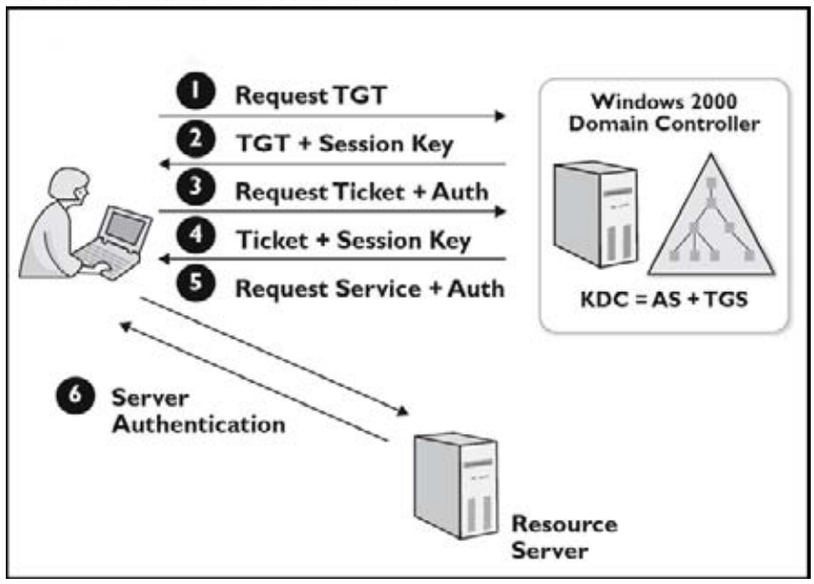


Figura 22: Resumen del funcionamiento de Kerberos

2.5.2.Configuración de Kerberos

Se debe editar el archivo /etc/krb5.conf y sus modificaciones son mínimas en relación al contenido del archivo, pues hay que sólo modificar la información del dominio Active Directory y el servidor de tickets (KDC).

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 ticket_lifetime = 24000
 default_realm = ANGLOCHILE.LOCAL
 dns_lookup_realm = false
 dns_lookup_kdc = false

[realms]
 ANGLOCHILE.LOCAL = {
  kdc = abclscl001.anglochile.local:88
  default_domain = ANGLOCHILE.LOCAL
 }

[domain_realm]
 .anglohile.local = ANGLOCHILE.LOCAL
 anglochile.local = ANGLOCHILE.LOCAL

[kdc]
 profile = /var/kerberos/krb5kdc/kdc.conf

[appdefaults]
 pam = {
   debug = false
   ticket_lifetime = 36000
   renew_lifetime = 36000
   forwardable = true
   krb4_convert = false
 }
```

CAPÍTULO 3

ESPECIFICACIÓN DE REQUISITOS

El objetivo de este capítulo es analizar las necesidades del negocio del cliente y establecer los requerimientos. La especificación de requisitos ha sido elaborada en conjunto con el cliente durante diversas reuniones para poder obtener:

- Requerimientos de la organización: estructura organizacional, políticas de recursos humanos y capacitación.
- Las necesidades del ambiente de la solución: necesidades del cliente, de seguridad, restricciones de espacio físico.
- Los requerimientos de calidad: facilidad de uso, mantención del sistema, compatibilidad, confiabilidad y capacidad de escalabilidad.
- Requerimientos generales de la solución: rendimiento, implementación, instalación y soporte operacional.
- Limitaciones: regulaciones de la organización, tiempo disponible, estándares.

3.1.Requerimientos generales

- Sólo se utilizarán paquetes de software certificados por el fabricante del Sistema Operativo con el propósito de eliminar fuentes de falla.
- Los equipos estarán instalados en un rack en el Data Center Corporativo de la Compañía.
- Se instalará como software de sistema operativo: WhiteBox Enterprise Linux 3.

3.2.Requerimientos de particiones de disco

El servidor cuenta con 2 discos en RAID 1 por hardware. Se necesita realizar una división de particiones, de tal forma que sea más sencillo realizar respaldos sobre cada una de las particiones y también con el fin de que la caché quede en una partición aislada, de tal forma que no interfiera con el funcionamiento del sistema.

Punto de montaje	Tipo de partición	Tamaño en MB
/boot	ext3	100
swap	swap	2xRAM
/usr	ext3	3000
/var	ext3	2000
/var/spool/squid	ext3	8000
/	ext3	2000

3.3.Requerimientos de servicios que deben poseer los servidores

Parte de la mejores prácticas (best practiques) en securitización de servidores consiste en desinstalar o deshabilitar los servicios que no son requeridos durante la operación del sistema, con el propósito de liberar los recursos utilizados y eliminar potenciales brechas de seguridad.

Los servicios mínimos requeridos para operar la plataforma se detallan a continuación:

- SSH: para el acceso remoto.
- NTP: para la sincronización del reloj de la máquina con un servidor NTP.
- Sendmail/Postfix: para el envío de reportes y mensajes a los administradores.
- CRON: para la ejecución de tareas en forma periódica.
- Squid: el servicio principal.
- Samba+Winbind+Kerberos: para la conexión con el servidor Active Directory.
- Heartbeat+LVS: herramienta de cluster.

3.4.Requerimientos de paquetes a instalar

Sólo se instalarán los paquetes mínimos necesarios que cumplan con los servicios solicitados en el punto anterior. No se considerará la instalación de paquetes de modo gráfico para no sobrecargar el servidor ni tampoco para utilizar mas espacio en el disco duro.

3.5.Monitoreo y alertas

EL proyecto suministrará al área de *Operaciones* las herramientas necesarias para identificar proactivamente los problemas que puedan ocurrir, en el ámbito de redes, sistema operativo y servicios. Por ejemplo, el sistema debe enviar un correo al área de Operaciones cada vez que se detenga el proceso “squid”, y lo mismo cuando quede poca cantidad de RAM y poco espacio libre en cada una de las particiones del disco duro.

3.5.1.Webmin Alert (CPU, particiones y memoria)

Los umbrales asociados a los parámetros antes indicados serán ajustados durante el proceso de puesta en marcha. Las alertas emitidas serán enviadas por correo a los operadores (chileoperadores@anglochile.cl) y a los administradores y supervisores del servicio (aomunoz@anglochile.cl, ramorgado@anglochile.cl).

3.5.2.Reporte por e-mail con resumen del estado diario de los servidores

La facilidad *LogWatch* -incluida en la instalación por defecto del servidor- será configurada para emitir un reporte diario a las 04:02 AM y enviada a los mismos destinatarios antes indicados.

3.5.3.Rotación y retención de logs en línea

Se hará uso de la herramienta *LogRotate* -incluida en la instalación por defecto del servidor- para inicializar diariamente a las 04:02 AM de cada día la rotación de los logs del sistema y los históricos se mantendrán hasta por 5 días atrás.

CAPÍTULO 4

ANÁLISIS Y DISEÑO DE LA SOLUCIÓN PROPUESTA

Este capítulo tiene por finalidad analizar diversas alternativas de solución, seleccionar una, documentar la solución propuesta y preparar un plan inicial del proyecto.

Se procederá a diseñar la solución de acuerdo a la especificación de requisitos descrita en el capítulo anterior. El diseño contempla:

- Seleccionar alternativas de solución: Solución Propuesta
- Especificar (describir) un diseño de solución
- Generar un plan de proyecto detallado

4.1.Análisis

En esta etapa se analizaron algunos productos y soluciones comerciales y no comerciales que se pueden utilizar para realizar una Propuesta de Solución. Los productos analizados han sido descritos en el punto 1.1.

Red Hat Cluster Suite (solución comercial de Red Hat, Inc.) ha sido descartado, pues, para poder instalarlo se necesita de un storage externo conectado hacia ambos servidores, lo cual aumenta considerablemente el costo de la solución, sin mencionar, además, el costo que tiene este producto.

Con respecto a **OpenMosix**, se debe señalar que la clusterización que realiza lo hace a un nivel muy bajo (a nivel de kernel), acercándose a las funciones de un sistema distribuido, en el cual se distribuye procesamiento en un grupo de máquinas, mientras que lo que se desea es balancear las conexiones de los clientes en tan solo

dos nodos.

El inconveniente que posee **Kimberlite** -para esta solución- es que no es muy popular. Es muy poco utilizado y se encuentra muy poca documentación acerca de este producto, por ende se encuentra muy poco soporte para la resolución de problemas que puedan ocurrir con este software.

Finalmente, **Ultramonkey** ofrece un cluster de alta disponibilidad y balanceo de carga soportando sin problemas hasta 200 nodos Slave. No se requiere la compra de licencias ni tiene limitante por cantidad de usuarios. Es un software libre, con licencia GPL (General Public Licence).

4.2.Solución propuesta

Tomando en consideración la especificación de requisitos, el análisis efectuado en el punto anterior, y a las topologías mostradas en el punto 2.3.8 – se ha llegado a la siguiente solución:

Alta disponibilidad y balanceo de carga simplificada

Esta topología combina las funciones de *Linux Director* y *Servidor Real* dentro de las mismas máquinas, así la alta disponibilidad y el balanceo de carga puede ser mantenido con sólo dos nodos.

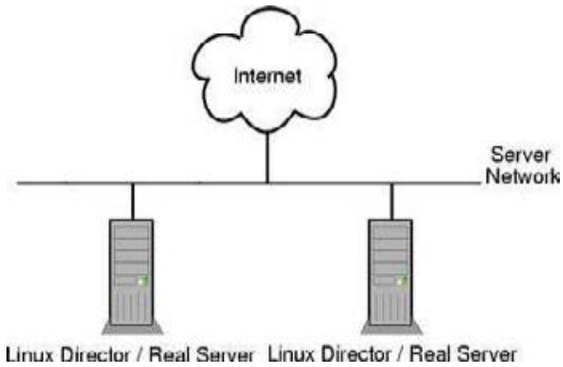


Figura 23: Alta disponibilidad y balanceo de carga simplificada

4.3.Descripción de la solución

4.3.1.Disponibilidad y balanceo del servicio Proxy

Se implementará una solución de carga balanceada y alta disponibilidad para el servicio Proxy utilizando un sistema de Servidores Virtuales (Ultramonkey), el cual es transparente para los usuarios, tanto para el ambiente de carga balanceada como para la contingencia de una falla (alta disponibilidad).

Para lograr lo anterior, los usuarios deben seguir “conectándose” hacia una misma dirección IP (Heartbeat) y debe existir un elemento que tome el requerimiento de conexión y lo derive por medio de algún algoritmo inteligente a uno de los dos servidores que prestan el servicio (LVS). Una de las cosas que aporta a la solución es que el servicio Proxy posee intrínsecamente elementos de colaboración (consultas de caché), lo que permite un mejor entendimiento a la hora de discriminar la carga de cada una de las máquinas para efectos de “quién” tomará el nuevo requerimiento.

Por lo tanto, para lograr la implementación de esta solución se requerirán 2 elementos. En primer lugar, deberá existir un nodo director (LVS) que tome el requerimiento y lo delegue entre los servidores que componen la granja de la solución, en segundo lugar un elemento que administre la dirección IP (Heartbeat) de conexión de los clientes que pueda “moverse” entre los servidores -en caso de falla de uno de ellos- para poder permitir la continuidad de operación de los usuarios.

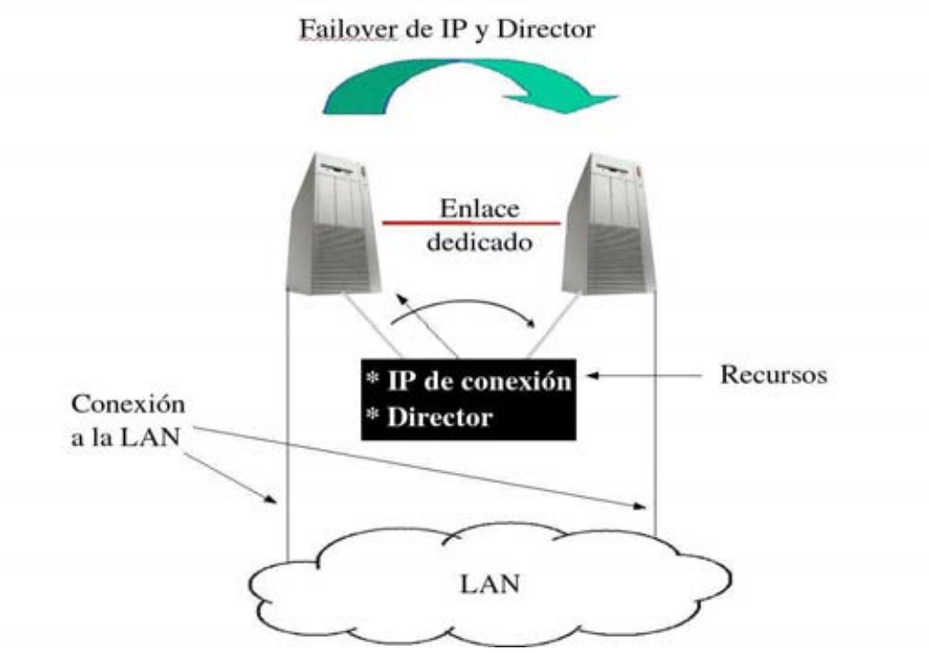


Figura 24: Esquema de la solución

4.3.2.Esquema de solución propuesta

En concreto, lo que se realizará es un esquema como el mostrado en la figura 24. el cual funcionalmente posee lo siguiente:

- Servidores con un sistema operativo actualizado: WhiteBox Linux 3
- Servicio Proxy: Squid
- Herramienta de administración de la dirección IP de conexión: Heartbeat
- Ambos con el “director” configurado (Ldirector), pero sólo en uno habilitado, de tal manera que en caso de falla del servidor activo tome su función el segundo servidor.

Con esta solución se garantiza la carga balanceada entre los dos servidores y la disponibilidad del servicio frente a la caída de uno de los nodos.

4.3.3.Conexión con Active Directory

Adicionalmente se desea que el acceso a Internet, por parte de los usuarios, se

realice por medio de autentificación contra el dominio Microsoft® de la organización, de tal manera que si el usuario posee permiso de navegación el browser no le solicite nuevamente sus credenciales. Para realizar este procedimiento cada servidor Proxy está configurado para conectarse al dominio Microsoft® Active Directory, traspasar las credenciales que le envió la estación de trabajo y realizar la consulta por perfiles de acceso, de tal manera de otorgar permiso de tráfico. Esta conexión entre Squid y Active Directory se logra mediante el uso de Samba, Winbind y Kerberos.

4.3.4.Registros y estadísticas de navegación

Una vez que se otorgue el permiso de tráfico cada usuario estará sujeto a las políticas de acceso definidas a través de SquidGuard, generándose los reportes de utilización por medio del software *SquidAnalyzer*, el cual se realizará por medio de un log unificado vía scripting, el cual toma los logs de cada servidor Proxy y los reúne en uno solo. Los reportes son visibles mediante una herramienta de administración vía Web, llamada *Webmin*, mediante la cual se han unificado los logs en una sola interfaz.

4.3.5.Respaldo y recuperación

Se generará un procedimiento de respaldo de los archivos de configuración y de los datos contra un servidor Linux en particular, para luego generar un procedimiento de recuperación parcial o total del sistema.

4.3.6.Requerimientos y alcances técnicos

La solución Linux es para un número de usuarios ilimitados, la única limitante son los recursos de hardware disponibles.

Se requiere hardware compatible con Linux, se asume que el cliente lo posee o lo adquirirá basada en las recomendaciones que se detallan a continuación:

- Se requieren servidores con dos tarjetas de red cada uno. Intel® EEPro es una de las NIC recomendadas con compatibilidad para la mayoría de las distribuciones de Linux.
- Se requieren servidores con al menos 2 GB de RAM (recomendable 4 GB) debido a la gran cantidad de recursos que utilizará el Proxy y la aplicación generadora de reportes de navegación.

4.4.Planificación del proyecto

Las tareas a realizar serán las siguientes:

- Instalación de servidores. Está compuesto por los siguientes servicios. Sujeto a los requerimientos antes descritos.
  - Instalar, actualizar y configurar dos servidores con el sistema operativo seleccionado.
  - Conectar los servidores a la red.
  - Instalar Proxy Squid en dos servidores.

- Balanceo de carga y alta disponibilidad de Squid. Está compuesto por los siguientes servicios. Sujeto a los requerimientos antes descritos.
  - Configuración básica de Squid en dos servidores.
  - Instalación de SquidGuard.
  - Instalación de herramienta generadora de reportes por usuario en dos servidores: SquidAnalyzer.
  - Generar script para unificar Logs de cada Proxy, de tal manera de unificar el tráfico en un único log, desde el cual se generará el reporte oficial de utilización con SquidAnalyzer.
  - Instalación de herramientas de replicación: Rsync.
  - Instalación de herramientas de carga balanceada: LVS.
  - Instalación de herramientas de administración de dirección IP de conexión y de servicios: Heartbeat.

- Configuración de ambiente de clusterización para permitir la conexión de los usuarios a los servidores y permitir la carga balanceada entre los dos servidores.
- Configuración del ambiente de alta disponibilidad para posibilitar la continuidad de servicio en casa de falla de uno de los nodos.

- Autenticación contra Dominio Microsoft. Está compuesto por los siguientes servicios. Sujeto a los requerimientos antes descritos.
  - Aprovechar la característica de Microsoft Internet Explorer de pasar las credenciales de autentificación de manera desatendida, no obligando al usuario a validarse nuevamente.
  - Traspasar a Squid la credencial que posibilita o no la utilización de Internet de cada usuario desde el repositorio Active Directory de Windows 2003 en ambos servidores.

- Monitoreo y alertas. Está compuesto por los siguientes servicios. Sujeto a los requerimientos antes descritos.
  - Instalación de herramienta Webmin, configurando alertas por correo cuando el servicio Squid no esté funcionando, o cuando quede poco espacio en el disco duro.

- Respaldo y recuperación. Está compuesto por los siguientes servicios.
  - Apoyo a la instalación de los agentes y clientes de *Data Protector* necesarios.
  - Definición de los archivos y datos a respaldar.
  - Generar un desastre en uno de los servidores. Chequear el correcto funcionamiento del cluster.
  - Recuperar el servidor. Instalar el sistema operativo, recuperar los archivos y datos de cinta, recuperar la configuración y dejar en funcionamiento la máquina. Conectar el servidor al cluster y dejar ambos nodos en funcionamiento.

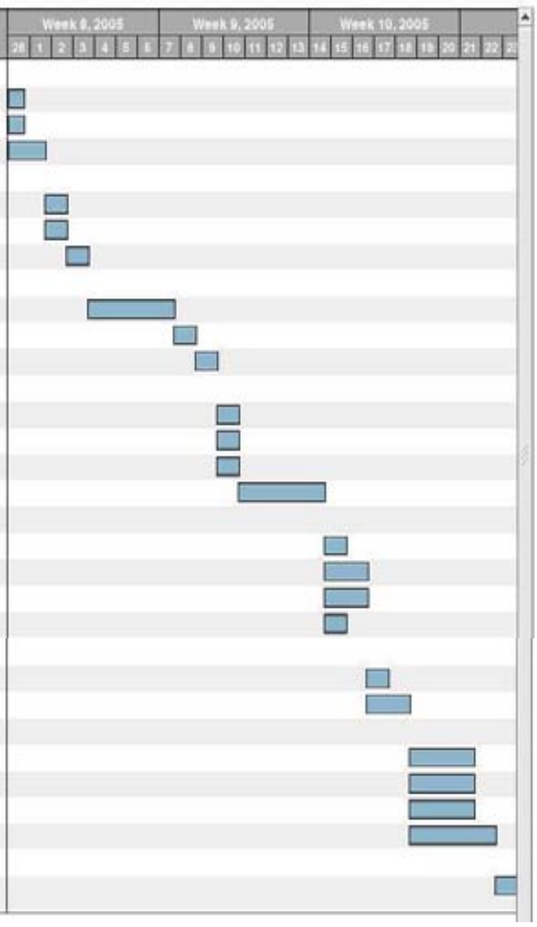
4.4.1.Carta Gantt

Proyecto Cluster Proxy

Company: Linux Center Latin America
Manager: Daniel Eugenin M.
Start Date: February 28, 2005
Finish Date: March 23, 2005
Duration: 23d 17h
Report Date: November 12, 2005

Gantt Chart

WBS	Name	Work
1	**Instalación de servidores**	**2d**
1.1	Instalación de 2 servidores	1d
1.2	Configuración de parámetros de red	1d
1.3	Actualización de 2 servidores	2d
2	**Configuración de Squid**	**2d**
2.1	Definición de políticas de navegación	1d
2.2	Configuración squid	1d
2.3	Instalación de SquidGuard	1d
3	**Balanceo de Carga y Alta Disponibilidad**	**4d**
3.1	Configuración LDirector	2d
3.2	Configuración HeartBeat	1d
3.3	Pruebas de failover	1d
4	**Autenticación contra dominio Microsoft**	**3d**
4.1	Configuración de Samba	1d
4.2	Configuración Kerberos	1d
4.3	Ingreso de los servidores al dominio	1d
4.4	Pruebas de autenticación	2d
5	**Monitoreo y alertas**	**2d**
5.1	Instalación de SquidAnalyzer	1d
5.2	Creación de script de consolidación de logs	2d
5.3	Instalación de Webmin	2d
5.4	Instalación agentes SNMP	1d
6	**Respaldo y recuperación**	**2d**
6.1	Instalación de agentes de Data Protector	1d
6.2	Generación de procedimiento de respaldo y recuperación	2d
7	**Pruebas finales**	**2d**
7.1	Definición de las pruebas a realizar	1d
7.2	Pruebas de failover	1d
7.3	Pruebas de autenticación de usuarios	1d
7.4	Pruebas de respaldo y recuperación	2d
8	**Puesta en producción**	**1d**
8.1	Soporte postventa	1d



CAPÍTULO 5

IMPLEMENTACION E INSTALACIÓN

En este capítulo se describen las principales tareas realizadas durante la implementación de la solución presentada.

5.1.Resumen de Actividades

- Instalación de servidores.
  - Instalación de dos servidores con WhiteBox Linux 3.
  - Conexión de los servidores a la red.
  - Actualización de los dos servidores.

- Configuración de Squid.
  - Instalación y configuración de Proxy Squid en ambos servidores.
  - Instalación y configuración de SquidGuard en ambos servidores.

- Balanceo de carga y alta disponibilidad de Squid.
  - Instalación de herramientas de carga balanceada: Ldirector.
  - Instalación de herramientas de administración de dirección IP de conexión y de servicios: Heartbeat.
  - Configuración del ambiente de alta disponibilidad para posibilitar la continuidad de servicio en caso de falla de uno de los nodos.

- Autenticación contra Dominio Microsoft.
  - Instalación de helper[3] Winbind en ambos servidores.

3 El helper permite a Squid conectarse a la mayoría de las bases de datos que posee PAM (entre ellos Active Directory) para realizar la autentificación de usuarios validando usuario y password.

- Instalación y configuración de Samba (conectado al PDC de la red) en ambos servidores.
- Conexión de Samba contra el servidor Active Directory.

- Respaldo y recuperación.
  - Apoyo a la instalación de los agentes y clientes de HP Data Protector® necesarios.
  - Instalación y configuración de Remote Shell (rsh), Remote Login (rlogin) y Remote Exec (rexec) para la utilización de sistema de respaldo HP Data Protector.
  - Definición de los archivos y datos a respaldar

- Herramientas de operación.
  - Instalación y configuración de herramienta visor de logs de Squid: SquidAnalyzer, en ambos servidores.
  - Instalación y configuración de Webmin con módulos de Administración: Webalizer y enlace hacia SquidAnalyzer.

La guía de los pasos de instalación y sus correspondientes archivos de configuración se encontrará en el ANEXO A de esta Tesis.

5.2.Respaldo y recuperación de un servidor

5.2.1.Procedimiento de respaldo

Requerimientos:

El servidor que se desee respaldar debe tener Remote login, Remote shell y Remote exec, para lo cual se debe chequear que el sistema posea el paquete ***rsh-server.***

Configuración:

La configuración del servidor para instalar los agentes de Data Protector es detallada en el ANEXO A.

A continuación, se pasa al procedimiento descrito en el punto 5.2.3 donde el usuario deberá utilizar el software de respaldo Data Protector para realizar el respaldo sobre una cinta.

5.2.2.Procedimiento de recuperación

A) Realizar una instalación básica en la máquina a recuperar:

Para esto se podrá utilizar un sistema llamado *Kickstart*, con el cual la máquina se instalará con unas opciones predefinidas sin que el usuario deba elegir algo durante la instalación.

Se recomienda tener guardado en un diskette el archivo /root/anaconda-ks.cfg existente en ambos servidores.

Iniciar la máquina a recuperar con el primer CD de instalación y en la primera pantalla de bienvenida se debe insertar el diskette y escribir lo siguiente:

linux ks=floppy

Con esto el sistema se instalará en forma automática y sólo pedirá los próximos CDs, sin que el usuario deba seleccionar alguna opción.

En caso de que el usuario no desee utilizar el sistema *Kickstart*, deberá instalar la máquina con las opciones por defecto.

B) Después que la máquina ha sido instalada, se deberá pasar al procedimiento anterior (Procedimiento de Respaldo, descrito en el punto 5.2.1).

C) Finalmente se pasa al procedimiento descrito en el punto 5.2.3 donde el usuario deberá utilizar el software Data Protector para realizar la recuperación de la máquina.

5.2.3.Utilización de software de respaldo

El cliente utilizará el software Data Protector para realizar esta labor guardándose un respaldo de cada servidor en cintas, con la posibilidad de realizar recovery (recuperación) cuando se necesite.

5.3.Herramientas de Monitoreo

5.3.1.SquidAnalyzer

SquidAnalyzer analiza el archivo de logs de Squid y genera un reporte de estadísticas acerca de los sitios (URLs) visitados por cada usuario, desplegando un calendario donde el usuario puede seleccionar las estadísticas acerca de un día en específico. Además se obtienen datos como los sitios mas visitados durante un día, los usuarios que mas navegan durante el día, etc. [Gilles, 2005]

Vistas de los reportes que se obtienen con SquidAnalyzer:

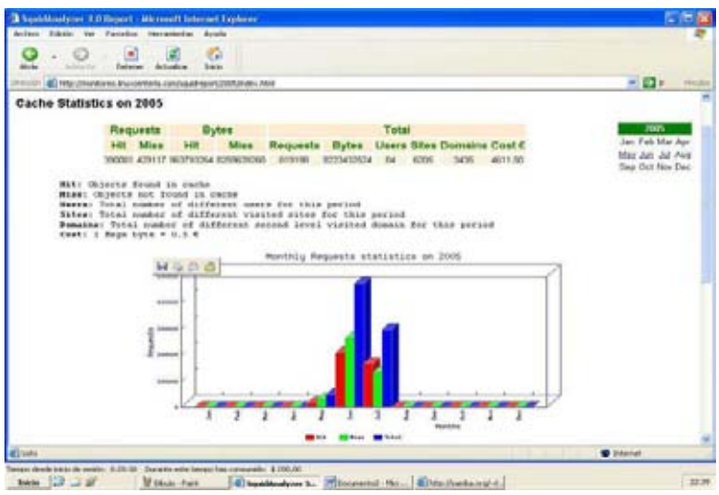


Figura 25: Vista general de SquidAnalyzer

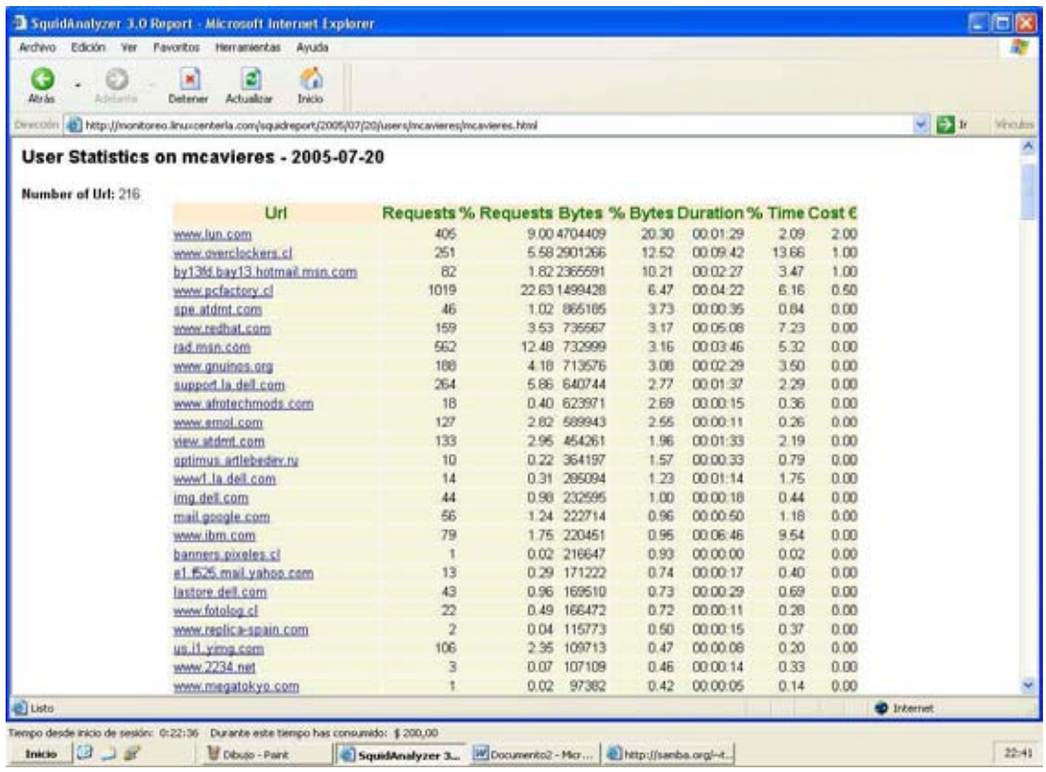


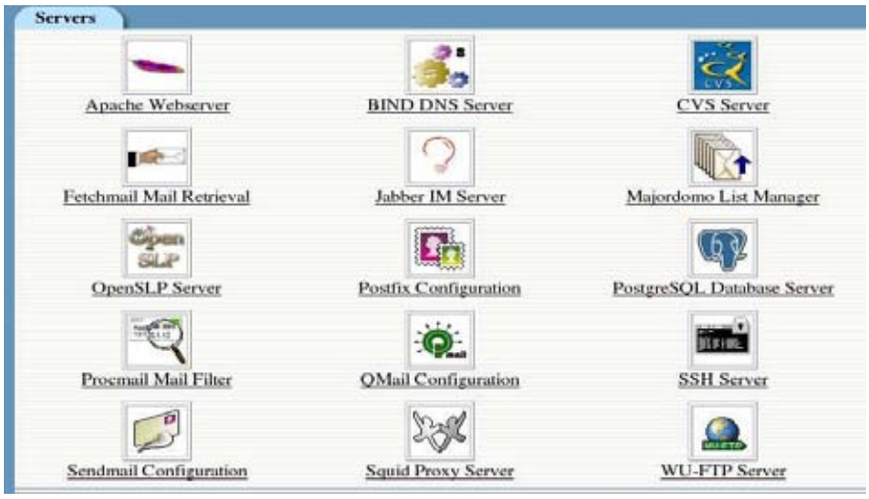
Figura 26: Vista de la navegación de un usuario en un día específico

5.3.2.Webmin

Webmin es una de las mejores herramientas de administración remota para un servidor Linux, escrita primariamente en Perl, fácil de usar y de instalar. Funciona en la mayoría de plataformas de Linux y en una variedad de otras plataformas UNIX. Lo más importante es que hace el sistema accesible a personas no técnicas, que tienen que administrar sistemas de tal forma que no se tenga que proporcionar cuentas reales en el servidor. [Cameron, 2005]

Vista de diversos servicios que se pueden controlar por medio de Webmin.

Figura 27: Vista general de Webmin



Vista de las restricciones de Squid y la posibilidad de modificar a gusto del usuario las reglas de filtro sobre máquinas.

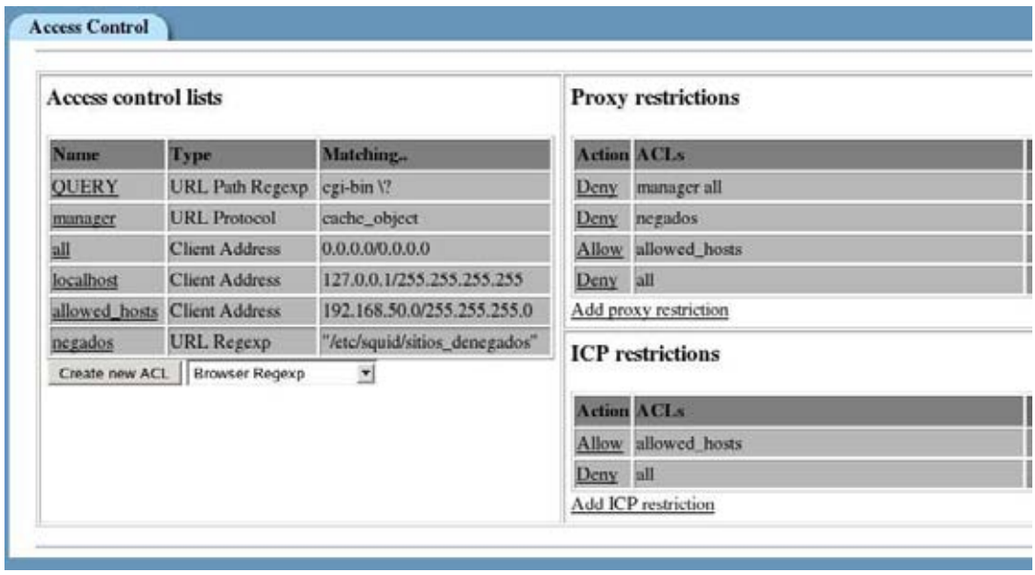


Figura 28: Control de reglas de Squid por medio de Webmin

CAPÍTULO 6

PUESTA EN PRODUCCION Y SOPORTE

Los objetivos de esta fase son:

- Distribuir e instalar todos los componentes de solución incluida.
- Conducir las pruebas del sistema.
- Conducir las pruebas de aceptación.
- Registrar, revisar y corregir la solución.
- Obtener la aceptación del cliente.
- Conducir el traspaso suave al cliente.
- Continuar la demanda de cambios.

Antes de la puesta en producción, el sistema debe pasar diversas pruebas definidas con anterioridad.

6.1.Pruebas del sistema

Las pruebas definidas contemplan ámbitos de autentificación, cluster y recuperación, por lo cual se han definido las siguientes pruebas:

6.1.1.Pruebas de navegación

6.1.1.1.Autentificación de usuarios

Estas pruebas se realizaron indicándole a los usuarios que testearon el sistema que utilicen la dirección IP de los servidores Proxy recién configurados e instalados para que naveguen por Internet.

Si a los usuarios no se les solicita las credenciales (login y password) quiere

decir que el sistema está logrando autentificar a los usuarios mediante el traspaso de credenciales hechas al servidor Active Directory al momento de autentificarse en la estación de trabajo.

6.1.1.2.Autorización de usuarios

También se le solicitó a un usuario -que no está dentro del grupo de los permitidos para navegar por Internet- que tratase de navegar, previamente configurado el Proxy en su browser, sin obtener resultado positivo, por lo cual se comprueba que el servidor Proxy está autentificando correctamente a los usuarios y autorizando sólo a aquellos que pertenecen al grupo permitido para navegar por Internet.

6.1.2.Pruebas de cluster

6.1.2.1.Desconexión de un nodo

Esta prueba consistió en desconectar uno de los nodos del cluster. Esto significa, apagar uno de los nodos mientras el cluster está funcionando.

En primer lugar se apagó el nodo “master” (el cual posee la IP flotante y está realizando el balanceo de carga). Se comprobó que a los pocos segundos se levantó automáticamente la IP flotante en el nodo secundario y es ahora este último nodo el que actúa como nodo director. Se volvió a prender el nodo principal y éste se integró automáticamente al cluster.

En segundo lugar se apagó el nodo secundario (el que no estaba realizando el balanceo) y se comprobó que todo continúa normalmente sin que el usuario detecte alguna anomalía.

6.1.2.2.Pruebas de balanceo

Estas pruebas consistieron en solicitarle a varios usuarios que naveguen por diferentes sitios Web, mientras se chequeaba en los logs de ambos servidores que se estuviera navegando normalmente.

6.1.3.Pruebas de “disaster and recovery” (desastre y recuperación)

Estas pruebas consistieron en suponer una falla de alguno de los servidores y se procedió a reinstalar utilizando el procedimiento de recuperación del sistema utilizando el software corporativo Data Protector. Esta recuperación se realizó sin inconvenientes y el sistema volvió a la normalidad.

6.2.Puesta en producción

La puesta en producción consistió en definir una fecha y hora para realizar la puesta en marcha del sistema.

La puesta en producción de las máquinas consistió en configurar las direcciones IP reales a utilizar, por lo cual sólo se tuvo que modificar la configuración de las tarjetas de red.

Posteriormente, el cliente traslada las máquinas desde la sala de laboratorio hacia el rack del Datacenter corporativo de la compañía. El cliente realiza todas las conexiones necesarias y se ponen en funcionamiento los servidores.

Finalmente, se realizan algunas pruebas de navegación, autentificación y autorización de usuarios y pruebas de cluster resultando todas las pruebas sin problemas.

CAPÍTULO 7

CONCLUSIONES

Debido a la existencia de numerosas aplicaciones de clustering, comerciales y no comerciales, es posible realizar un análisis mas detallado en cuanto a la solución final a ofrecer, pudiendo adaptar alguna de estas aplicaciones a las necesidades específicas que requiere la solución.

Squid corresponde al software de Proxy mas popular existente en Internet, es robusto, estable y altamente configurable. Además, gracias a su integración con otras aplicaciones, es posible instalar un servidor Proxy en prácticamente cualquier ambiente de trabajo, satisfaciendo todas las posibilidades de filtraje que se necesite.

Los cluster LVS (Linux Virtual Servers) proveen balanceo de carga y alta disponibilidad para servicios de red a un bajo costo. Las soluciones no requieren modificaciones de cualquiera, los clientes o los servidores; y ellos deben soportar en su mayoría servicios TCP o UDP. Un balanceador de carga LVS (director) es diseñado para manejar millones de conexiones concurrentes. Además, es fácil configurar servicios de red altamente disponibles y altamente escalables utilizando cluster LVS. Comparado con otros productos comerciales, LVS provee muchas características únicas:

- Soporte de tres tecnologías de balanceo de carga IP, donde VS/TUN (Tunneling) y VS/DR (Direct Routing) proveen un muy alto rendimiento.
- Soporte de múltiples algoritmos de balanceo para despachar las conexiones a los servidores reales. Adicionalmente, algoritmos personalizados pueden ser agregados como módulos extras.
- Ofrece muchas -flexibles y extensibles- herramientas de monitoreo de servicios para mantener la alta disponibilidad de varios sistemas.

- Un código base robusto y estable, una base de usuarios y desarrolladores, una comunidad activa de usuarios, y la madurez del código que provee la revisión alrededor del mundo.
- Fiabilidad comprobada en aplicaciones del mundo real.
- Lo mejor de todo, LVS es libre para todo el mundo.

Con la gran cantidad de herramientas libres de monitoreo que existen en plataforma Linux, es posible construir un completo sistema para monitorear diversos aspectos de los servidores, ya sea a nivel de servicios (puertos, protocolos, etc.), como a nivel de Sistema Operativo (espacio en disco, memoria disponible, etc.); todo esto a un costo muy bajo, lo cual ayuda a reducir los costos de implementación.

Gracias a la utilización de poderosas herramientas de respaldo y recuperación -como es el caso de Data Protector- es muy sencillo recuperar un servidor cuando éste ha sufrido algún desperfecto, pues tan sólo basta programar un respaldo periódico en cintas olvidándose de esta tarea. Mientras que la recuperación de un servidor tan sólo se limita a un simple click dentro de la aplicación.

Finalmente, con el uso de aplicaciones de software libre, instalados en un Sistema Operativo Linux, es posible reducir considerablemente los costos de implementación de soluciones reduciéndolo sólo a costos de ingeniería involucrados en el proyecto. Además, el tiempo de implementación se ve reducido gracias a la enorme cantidad de documentación que es posible encontrar en Internet.

APÉNDICE A

Glosario

Este glosario define términos y acrónimos que se utilizan en esta Tesis. Los términos con varias palabras están ordenados alfabéticamente por el nombre principal del término.

ACK. El indicador de TCP que confirma la recepción de un segmento TCP recibido anteriormente.

Administración proactiva. Involucra las actividades planificadas para asegurar que la red cumpla con los cambios en la demanda, manteniendo los costos lo más bajo posible.

Administración reactiva. Involucra la detección y resolución rápida de un problema que ocurre en la red.

Autenticación. Es el proceso de determinar que una entidad es quien o lo que dice ser.

Autorización. Es el proceso de determinar los recursos y servicios que puede usar una entidad.

Broadcast. Ver *Transmisión por difusión*.

Cortafuego. Ver *Firewall*.

CRON. Es un sistema demonio llamado crond y unos archivos de configuración y secuencias de comandos que ponen en marcha tareas del sistema programadas.

Demonio. Un servidor de servicios de sistema básicos que se ejecuta en segundo plano.

Dirección IP. Es un identificador numérico único que se asigna a una red específica o a una interfaz de red de un dispositivo en una red. Es una dirección de software que se puede traducir directamente a un host o nombre de red comprensible por el usuario. Las direcciones IP de interfaz de red de host también se asocian con una o más direcciones de interfaz de red de hardware.

Enmascaramiento. Proceso de sustituir una dirección origen local en un paquete saliente con la del firewall o máquina que hace de pasarela, de forma que permanezcan ocultas las direcciones IP de la LAN. El paquete parece proceder de la máquina de pasarela en lugar de una máquina interna de la LAN. El proceso se invierte para paquetes de respuesta entrantes desde servidores remotos. La dirección de destino del paquete, la dirección IP de la máquina firewall, se sustituye con la dirección de la máquina cliente dentro de la LAN interna. El enmascaramiento IP se suele llamar traducción de direcciones de red (NAT, Network Address Translation).

Firewall. Es un nodo configurado como una barrera para impedir que el tráfico cruce de una red a otra. Los firewalls se utilizan como medida de seguridad para aislar subredes y/o redes privadas de redes públicas.

Gratuitous ARP. Un ARP no pedido es una respuesta ARP cuando no hubo una petición ARP.

HTTP. HiperText Transfer Protocol (Protocolo de transferencia de hipertexto). Se utiliza en servidores y navegadores web.

Hub. Es un dispositivo que se utiliza en una red y que sirve de emplazamiento central para conectar las estaciones de trabajo en sí. El término Hub se utiliza a menudo como término genérico para denominar a un componente de equipo de red.

ICMP. (Internet Control Message Protocol). El Protocolo de Control de Mensajes Internet permite el envío de mensajes de control y error entre distintos gateways, routers o hosts.

Interfaz de red. Tarjeta o adaptador que permite conectar un equipo a una red.

IP. (Internet Protocol). IP es un protocolo de la capa de red en el modelo de referencia TCP/IP, el cual es un protocolo estándar para enviar una unidad de datos básica (un datagrama IP) a través de una interconexión de redes. IP representa el esquema por el cual dos dispositivos (ambos con direcciones IP) se comunican entre sí.

ISP. Proveedor de acceso a Internet. Persona natural o jurídica que presta el servicio de acceso a Internet, de conformidad a la ley y su normativa complementaria.

ISO. (International Standard Organization). Organización International de Estándares.

LAN. Local Area Network. Red de área local. Es una red de dispositivos conectados (como PCs, impresoras, servidores y hubs) que cubren un área geográfica relativamente pequeña (generalmente no más grande que una planta o un edificio). Las LANs se caracterizan por transmisiones de alta velocidad en cortas distancias. Ethernet, FDDI y Token Ring son tecnologías ampliamente utilizadas en la configuración de LANs.

Linux Director. Máquina con Linux y LVS instalado, el cual recibe paquetes desde los usuarios finales y los envía a los servidores reales.

Modelo de red OSI. (Open System Interconnection). Modelo de referencia de Interconexión de Sistemas desarrollado por la ISO con el fin de crear estándares comunes de comunicación entre equipos de diferentes fabricantes.

PDC. (Primary Domain Controller). Controlador Primario de Dominio.

SCSI. (Small Computer System Interface). Estándar para conectar dispositivos periféricos a los computadores, permitiendo altas tasas de transmición de datos.

Servidor Real. Máquina en donde termina la conexión. Éste estará ejecutando algún demonio tal como Apache.

Real IP Address. Es la dirección IP de un Servidor Real.

SMB. (Server Message Block). Es el formato de los mensajes utilizado por los sistemas operativos tipo DOS y Windows para compartir archivos, directorios y dispositivos. NetBIOS está basado en el formato SMB.

SNMP. (Simple Network Management Protocol). El Protocolo Simple de Administración de Red es un protocolo diseñado para dar al usuario la capacidad de administrar remotamente dispositivos de red.

TCP. (Transfer Control Protocol), Es un protocolo de la capa de transporte en el modelo de referencia TCP/IP, el cual se encarga de administrar el flujo de paquetes IP, garantizando que éstos lleguen a su destino correctamente y libres de errores.

TCP/IP. (Transfer Control Protocol/Internet Protocol). Corresponde al modelo de referencia de red desarrollado por el Departamento de Defensa de los Estados Unidos en la década de los años 70, para permitir la comunicación entre el equipo de diferentes fabricantes.

Transmisión por difusión (broadcast). Un paquete de datos enviado por un dispositivo a todos los nodos de la red.

Usuario final. Máquina que origina una conexión.

Virtual IP Address. Es la dirección IP asignada a un servicio que el Linux Director manejará.

WAN. (Wide Area Network). Es una interconexión de dispositivos que cubren un área geográfica grande (ciudades, países, continentes).

WINS. Windows Internet Naming Service.

BIBLIOGRAFIA

Baltzersen, Paul; Haland, Lars. *SquidGuard*. ONLINE.
<http://www.squidguard.org/>. Revisado 11 de diciembre de 2004.

Cameron, Jamie. *Webmin*. ON LINE.
<http://www.webmin.com>. Revisado 24 de octubre del 2005.

Chadd, Adrian. *Squid-Cache Project Page*. ONLINE.
<http://www.squid-cache.org/>. Revisado 16 de octubre de 2004.

Gilles, Darold. *SquidAnalyzer*. ONLINE.
<http://themes.freshmeat.net/projects/squidanalyzer/>. Revisado 20 de enero de 2005.

Gómez, Virgilio. *Samba*. Sólo Linux. Año 2. Número 11. España. 2000. 82 p.

Hatch, Brian; Lee, James; Kurtz, George. *Hackers en Linux*. McGraw-Hill. Madrid. 2001. 624 p.

Horman, Simon. *Ultramonkey Project Page*. ONLINE.
<http://www.ultramonkey.org/>. Revisado 21 de noviembre de 2004.

Jara, Andrés; Pintado, Juan; Serrano, Marcelo; Ullauri, Raúl; Juca, Angel. *Configurando un servidor Proxy utilizando Squid*. ONLINE.
<http://www.uazuay.edu.ec/estudios/sistemas/teleproceso/ciclo2/proxy.doc/>. Revisado 6 de enero de 2005.

Kimberlite Team. ONLINE.
<http://oss.missioncriticallinux.com/projects/kimberlite/>. Revisado 10 de enero de 2005.

Lucas, Nestor. *Kerberos*. Linux Actual. Año 1. Número 14. España. 2000. 82 p.

Microsoft Corporation. *Active Directory*. ONLINE. <http://www.microsoft.com/windowsserver2003/technologies/directory/activedirectory/default.mspx>. Revisado 9 de enero de 2005.

Moshe, Bar. *The OpenMosix Project Page*. ONLINE. <http://openmosix.sourceforge.net/>. Revisado 15 de octubre de 2004.

Paredes, Juan. *Soluciones HA para Linux*. ONLINE. <http://es.tldp.org/Presentaciones/200103hispaLinux/paredes/html/x135.html>. Revisado 7 de noviembre de 2004.

Red Hat, Inc. *Red Hat Cluster Suite*. ONLINE. <http://www.redhat.com/software/rha/cluster/>. Revisado 5 de enero de 2005.

Robertson, Alan. *Linux High Availability*. ONLINE. <http://linux-ha.org>. Revisado 12 de diciembre de 2004.

Samba Team. *Samba Project Page*. ONLINE. <http://www.samba.org/>. Revisado 14 de noviembre de 2004.

Zhang, Wensong. *Linux Virtual Servers*. ONLINE. <http://linuxvirtualserver.org>. Revisado 14 de septiembre de 2005.

Zhang, Wensong; Zhang, Wenzhuo. *Cluster*. ONLINE. <http://www.linux-mag.com/2003-11/clusters_01.html>. Revisado 16 de enero de 2005.

Ziegler, Robert L. *Firewalls Linux, Guía Avanzada*. Prentice Hall. Madrid. 2000. 474 p.

ANEXO

Guía de Instalación y Configuración

Guía de Instalación y Configuración

1. Instalación del sistema

A continuación se detalla archivo /root/anaconda-ks.cfg con todas las opciones seleccionadas durante la instalación.

Este archivo es creado automáticamente al finalizar la instalación de una máquina, el cual se puede respaldar en algún medio para posteriormente utilizarlo en la instalación de una máquina con las mismas opciones de instalación que se seleccionaron en un principio.

Archivo de instalación de los servidores:

```
install
cdrom
lang en_US.UTF-8
langsupport --default=en_US.UTF-8 en_US.UTF-8 es_CL.UTF-8
keyboard la-latin1
xconfig --card "RIVA TNT2" --videoram 16320 --hsync 30-54 --vsync 50-120 --
resolution 800x600 --depth 16 --startxonboot  --defaultdesktop gnome
network --device eth0 --bootproto dhcp
rootpw --iscrypted $1$fr6zMk1o$Z0YPBwGfoYO9UJgyxlVI7/
firewall --disabled
selinux --disabled
authconfig --enableshadow --enablemd5
timezone --utc America/Santiago
bootloader --location=mbr --append="rhgb quiet"

part /boot --fstype "ext3" --size=100 --ondisk=hda
part pv.3 --size=0 --grow --ondisk=hda
volgroup VolGroup00 --pesize=32768 pv.3
logvol / --fstype ext3 --name=LogVol00 --vgname=VolGroup00 --size=1024 --grow
logvol swap --fstype swap --name=LogVol01 --vgname=VolGroup00 --size=512 --
grow --maxsize=1024

%packages
@ admin-tools
@ editors
@ system-tools
@ text-internet
@ ftp-server
@ smb-server
@ base-x
@ printing
@ spanish-support
kernel
lvm2
grub
e2fsprogs

%post
```

2. Sincronización con servidor NTP (Network Time Protocol)

La sincronización contra el servidor NTP de la compañía se realizó mediante un CRON que se ejecuta cada una hora. Esto se configuró en ambos servidores, en el archivo /etc/crontab insertando la siguiente línea al final del archivo:

```
5 * * * * root ntpdate -v -s ntp.anglochile.local
```

Luego se procede a reiniciar el servicio CRON para que tome los cambios. Para esto, se utiliza el siguiente comando:

$service crond restart

Con esta configuración se asegura que cada uno de los servidores tendrán su reloj del sistema sincronizado, pues es de gran importancia la hora de los servidores Proxy con respecto a la hora que posea el servidor Active Directory, ya que, si son distintas, existirán problemas de autentificación de usuarios al momento de utilizar Kerberos.

3. Configuración de Samba+Winbind

La configuración de Samba, junto a Winbind, se realiza en un único archivo de configuración. En ambas máquinas se edita el archivo /etc/samba/smb.conf.

Archivo de configuración de los servidores:

```
#=================== Global Settings ========================
[global]
   workgroup = SANCMD
   netbios name = sgolnx04
   server string = Proxy 1

   log file = /var/log/samba/%m.log
   max log size = 0

   security = ads
   realm = ANGLOCHILE.LOCAL
;   password server = *
   password server = abclscl001.anglochile.local
   encrypt passwords = yes

   socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192

   local master = no
   domain master = no
   preferred master = no

   wins server = 172.26.192.54
   dns proxy = no

# winbind Configuration
   winbind separator = +
   winbind uid = 10000-20000
   winbind gid = 10000-20000
   winbind enum users = yes
   winbind enum groups = yes
   winbind use default domain = yes
```

Luego se procede a iniciar el servicio SMB para que tome los cambios. Para esto, se utiliza el siguiente comando:

$service smb start

$chkconfig smb on

4. Configuración de Kerberos

En la configuración de Kerberos se define cuál es el dominio a utilizar (ANGLOCHILE.LOCAL) y cuál es el servidor Kerberos para realizar la autentificación de los usuarios (abclscl001.anglochile.local).

En primer lugar, se debe chequear que los servidores posean el paquete *krb5-libs* instalado, utilizando el comando: *$rpm -q krb5-libs*

En ambos servidores se edita el archivo /etc/krb5.conf:

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 ticket_lifetime = 24000
 default_realm = ANGLOCHILE.LOCAL
 dns_lookup_realm = false
 dns_lookup_kdc = false

[realms]
 ANGLOCHILE.LOCAL = {
  kdc = abclscl001.anglochile.local:88
  default_domain = ANGLOCHILE.LOCAL
 }

[domain_realm]
 .anglohile.local = ANGLOCHILE.LOCAL
 anglochile.local = ANGLOCHILE.LOCAL

[kdc]
 profile = /var/kerberos/krb5kdc/kdc.conf

[appdefaults]
 pam = {
   debug = false
   ticket_lifetime = 36000
   renew_lifetime = 36000
   forwardable = true
   krb4_convert = false
 }
```

4.1. Ingreso de la máquina al dominio Active Directory:

Cada uno de los servidores se ingresan al dominio de Active Directory con el siguiente comando: *$net ads join admin%DOMAIN*

Finalmente, se debe chequear que cada uno de los servidores pueda “conversar” con el servidor Active Directory, escribiendo los siguientes comandos:

$wbinfo -u <- debería entregar un listado de los usuarios del dominio

$wbinfo -g <- debería entregar un listado de los grupos existentes en el dominio

5. Configuración de Squid

Archivo de configuración principal, en el cual se define lo siguiente:

- Puerto en que “escuchará” el Proxy (8080)
- Tamaño de la memoria RAM a utilizar (2 GB)
- Tamaño máximo de la caché a guardar (10 GB)
- Conexión con listas negras (RBL) de sitios Web (SquidGuard)
- Modalidad de *sibiling* para consultar la caché que posee el servidor hermano
- Utilización y cantidad de procesos “helper” para la autentificación de usuarios
- Definición de ACL “Access Control Lists” para permitir la navegación

En primer lugar se debe chequear que los servidores posean el paquete *squid* instalado, utilizando el siguiente comando: *$rpm -q squid*

Se edita el archivo de configuración de Squid: /etc/squid/squid.conf

```
http_port 8080
hierarchy_stoplist cgi-bin ?
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY
cache_mem 1000 MB
cache_dir ufs /var/spool/squid 10000 16 256
cache_access_log /var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log none
redirect_program /usr/bin/squidguard -c /etc/squid/squidguard.conf
redirect_children 20
auth_param ntlm program /usr/bin/ntlm_auth --helper-protocol=squid-2.5-
ntlmssp
auth_param ntlm children 50
auth_param ntlm max_challenge_reuses 0
auth_param ntlm max_challenge_lifetime 5 minutes
auth_param basic program /usr/bin/ntlm_auth --helper-protocol=squid-2.5-
ntlmssp
auth_param basic children 50
auth_param basic realm Squid Proxy2
auth_param basic credentialsttl 2 hours
external_acl_type nt_group ttl=3600 concurrency=20 %LOGIN /
usr/lib/squid/wbinfo_group.pl
refresh_pattern ^ftp:           1440   20%    10080
refresh_pattern ^gopher:  1440   0%     1440
refresh_pattern .         0      20%    4320
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl to_localhost dst 127.0.0.0/8
acl SSL_ports port 443 563
```

```
acl Safe_ports port 80          # http
acl Safe_ports port 81          # http
acl Safe_ports port 20          # ftp-data
acl Safe_ports port 21          # ftp
acl Safe_ports port 443 563     # https, snews
acl Safe_ports port 70          # gopher
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
acl Safe_ports port 591         # filemaker
acl Safe_ports port 777         # multiling http
acl CONNECT method CONNECT
http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
acl dominios dstdomain "/etc/squid/dominios.txt"
acl Authenticated external nt_group SGO.WWW.USERS.GG
acl usuarios proxy_auth REQUIRED
acl ex-bodenor src 172.26.211.178
acl skype src 172.26.193.203
acl hotmail_domains dstdomain .hotmail.msn.com
acl ie6 browser MSIE[[:space:]]6
acl archivos urlpath_regex [-i] \.gif$ \.css$ \.zip$ \.class$ \.js$ \.jar$
acl FTP proto ftp
http_access allow dominios
http_access allow ex-bodenor
http_access allow skype
http_access allow archivos
http_access allow FTP
always_direct allow FTP
header_access Accept-Encoding deny ie6 hotmail_domains
http_access allow Authenticated
http_access allow usuarios
http_access allow localhost
http_access deny all
http_reply_access allow all
icp_access allow all
coredump_dir /var/spool/squid
redirect_program /usr/bin/squidguard -c /etc/squid/squidguard.conf
```

Luego se procede a iniciar el servicio SQUID para que tome los cambios. Para esto, se utiliza el siguiente comando:

$service squid start

$chkconfig squid on

6. Configuración de SquidGuard

Para utilizar esta aplicación se debe instalar los siguientes paquetes:

- squidguard
- squidguard-blacklists

Estos paquetes no son parte de la distribución, por lo cual se deben descargar desde Internet en la siguiente dirección:

http://apt.sw.be/redhat/el3/en/i386/RPMS.dag/

La descarga se puede realizar con los siguientes comandos:

$wget http://apt.sw.be/redhat/el3/en/i386/RPMS.dag/squidguard-1.2.0-2.rhel3.dag.i386.rpm

$wget http://apt.sw.be/redhat/el3/en/i386/RPMS.dag/squidguard-blacklists-20050528-1.1.el3.rf.noarch.rpm

La instalación del paquete se puede realizar con el siguiente comando:

$rpm -ivh squidguard.rpm*

Archivo de configuración /etc/squid/squidguard.conf:

```
# CONFIGURATION DIRECTORIES

dbhome /var/lib/squidguard
logdir /var/log/squidguard

# SOURCE ADDRESSES:

src lansource {
    ip 172.16.0.0/12 192.168.0.0/24 192.168.1.0/24
}

# DESTINATION CLASSES:

dest entertainment {
    domainlist entertainment/domains
    urllist entertainment/urls
}

dest local {
    domainlist local/domains
}

#expressionlist good/expressions
dest good {
    domainlist good/domains
    urllist good/urls
}
```

```
dest virus_exceptions {
    urllist virus_exceptions/urls
}

dest porn {
    domainlist porn/domains
    urllist porn/urls
    expressionlist  porn/expressions
    log porn.log
}

dest adult {
    domainlist adult/domains
    urllist adult/urls
    expressionlist  adult/expressions
}

dest audio-video {
    domainlist audio-video/domains
    urllist audio-video/urls
    log audio-video.log
}

dest forums {
    domainlist forums/domains
    urllist forums/urls
    expressionlist  forums/expressions
}

dest hacking {
    domainlist hacking/domains
    urllist hacking/urls
}

dest redirector {
    domainlist redirector/domains
    urllist redirector/urls
    expressionlist  redirector/expressions
}

dest warez {
    domainlist warez/domains
    urllist warez/urls
}

dest ads {
    domainlist ads/domains
    urllist ads/urls
}

dest aggressive {
    domainlist aggressive/domains
    urllist aggressive/urls
}

dest drugs {
    domainlist drugs/domains
    urllist drugs/urls
}

dest gambling {
    domainlist gambling/domains
    urllist gambling/urls
}

dest publicite {
    domainlist publicite/domains
    urllist publicite/urls
    expressionlist  publicite/expressions
}

dest violence {
    domainlist violence/domains
    urllist violence/urls
    expressionlist  violence/expressions
}
```

```
dest banneddestination {
    domainlist banneddestination/domains
    urllist banneddestination/urls
    expressionlist banneddestination/expressions
}

dest advertising {
    domainlist advertising/domains
    urllist advertising/urls
    redirect http://proxy.anglochile.cl/cgi-bin/nulbanner.png
    log advertising.log
}


### ACLs ###

acl {
    lansource {
        pass good local !porn !adult !audio-video !forums !hacking !redirector !warez
 ads !aggressive !drugs !gambling !publicite !violence !banneddestination !
advertising !entertainment any
        redirect http://proxy.anglochile.cl/cgi-bin/AccesoDenegado.cgi?clientaddr=%
a&srcclass=%s&targetclass=%t&clientname=%i&url=%u
    }
    default {
       pass local none
        redirect http://proxy.anglochile.cl/cgi-bin/AccesoDenegado.cgi?clientaddr=%
a&srcclass=%s&targetclass=%t&clientname=%i&url=%u
    }
}
```

Luego se procede a reiniciar el servicio SQUID para que tome los cambios de conexión de Squid con SquidGuard. Para esto, se utiliza el siguiente comando:

$service squid restart

7. Configuración de Ultramonkey

En primer lugar, se debe descargar todos los paquetes necesarios para la instalación de Heartbeat y Ldirector. Estos paquetes no son parte de la distribución, por lo cual se pueden descargar desde el sitio:

http://www.ultramonkey.org/download/rhel3/RPMS

La descarga de los paquetes se puede realizar con el siguiente comando:

*$wget -c -b -r http://www.ultramonkey.org/download/rhel3/RPMS/**

La instalación de los paquetes se puede realizar con los siguientes comandos:

$rpm -ivh perl.rpm*

$rpm -ivh ipvsadm.rpm*

$rpm -ivh heartbeat.rpm*

La configuración del cluster se realiza en ambos nodos por igual.

/etc/ha.d/authkeys:

```
auth 2
2 sha1 ultramonkeys
```

/etc/ha.d/ha.cf:

```
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility     local0
keepalive 2
deadtime 15
warntime 10
initdead 35
nice_failback on
bcast eth1
node    sgolnx04.anglochile.local
node    sgolnx05.anglochile.local
```

/etc/ha.d/haresources:

```
sgolnx04.anglochile.cl Ipaddr::172.26.192.20/16/eth0 \
                                        ldirectord::/etc/ha.d/ldirectord.cf
```

/etc/ha.d/ldirectord.cf:

```
checktimeout=10
checkinterval=2
autoreload=no
logfile="/var/log/ldirectord.log"
quiescent=yes
virtual=172.26.192.20:8080
        real=172.26.192.22:8080 gate
        real=172.26.192.23:8080 gate
        checktype=connect
        protocol=tcp
        checkport=8080
        service=none
        scheduler=wlc
```

Luego se procede a iniciar el servicio HEARTBEAT para que tome los cambios. Para esto, se utiliza el siguiente comando:

$service heartbeat start

$chkconfig heartbeat on

$chkconfig ldirectord off

8. Instalación y configuración de herramientas de monitoreo

En primer lugar, se debe instalar el paquete *Webmin*, el cual no forma parte de la distribución, por lo cual se descarga desde:

http://apt.sw.be/redhat/el3/en/i386/RPMS.dag/

La descarga se puede realizar con el siguiente comando:
$wget -c http://apt.sw.be/redhat/el3/en/i386/RPMS.dag/webmin-1.220-1.1.el3.rf.noarch.rpm

Instalación del paquete se puede realizar con el siguiente comando:

$rpm -ivh webmin-1.220-1.1.el3.rf.noarch.rpm

Después de instalada la aplicación, no se requiere configuración alguna, por lo que puede ser accedida a través de su sitio Web seguro creado en el puerto 10000.

https://sgolnx04.anglochile.local:10000

https://sgolnx05.anglochile.local:10000

9. Configuración de servidor para agentes de Data Protector

Esta configuración sólo se realiza en uno de los nodos (el que está conectado a la unidad de cinta).

El primer paso es verificar que el servidor posea el paquete *rsh-server* instalado, utilizando el siguiente comando:

$rpm -q rsh-server

Editar el archivo /etc/xinetd.d/rlogin

```
service login
{
        disable                 = no
        socket_type             = stream
        wait                    = no
        user                    = root
        log_on_success          += USERID
        log_on_failure          += USERID
        server                  = /usr/sbin/in.rlogind
        server_args             = -h
}
```

Editar el archivo /etc/xinetd.d/rsh

```
service shell
{
        disable                 = no
        socket_type             = stream
        wait                    = no
        user                    = root
        log_on_success          += USERID
        log_on_failure          += USERID
        server                  = /usr/sbin/in.rshd
        server_args             = -L -h
}
```

Editar el archivo /etc/xinetd.d/rexec

```
service exec
{
        disable                 = no
        socket_type             = stream
        wait                    = no
        user                    = root
        log_on_success          += USERID
        log_on_failure          += USERID
        server                  = /usr/sbin/in.rexecd
        server_args             = -h
}
```

Crear el archivo oculto /root/.rhosts, el cual debe contener el nombre del servidor de Data Protector:

```
sgoora04.anglochile.cl root
```

Finalmente ejecutar el comando:

$service xinetd restart

Con esto el usuario ya puede utilizar el software de respaldo Data Protector para conectarse a la máquina y realizar el respaldo en una cinta.

Consideración:

El módulo (driver) que maneja la tarjeta SCSI para respaldo se llama “aic7xxx”, se puede verificar que el módulo se esté ejecutando, utilizando el siguiente comando:

$lsmod | grep aic7xxx

Si no hay resultados, el módulo se puede cargar con el comando:

$modprobe aic7xxx (la ejecución de este comando demora algunos segundos).

Este comando se puede dejar dentro del script de inicio rc.local para que se

ejecute cada vez que se inicie la máquina. Para lo cual hay que editar el archivo / etc/rc.d/rc.local y agregar:

```
modprobe aic7xxx
```