DISEÑO E IMPLEMENTACIÓN DE UNA INFRAESTRUCTURA PARA LA PRESTACIÓN DEL SERVICIO DE CORREO USANDO LA TÉCNICA DE CLUSTERING DE ALTA DISPONIBILIDAD

MARIO JULIÁN BONILLA CONTRERAS

LUIS EDUARDO VARGAS BECERRA

INGENIERÍA DE SISTEMAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA, 2009

DISEÑO E IMPLEMENTACIÓN DE UNA INFRAESTRUCTURA PARA LA PRESTACIÓN DEL SERVICIO DE CORREO USANDO LA TÉCNICA DE CLUSTERING DE ALTA DISPONIBILIDAD

AUTORES
MARIO JULIÁN BONILLA CONTRERAS
LUIS EDUARDO VARGAS BECERRA

Trabajo de grado para optar por el título de
Ingeniero de sistemas

DIRECTOR
MPE. HENRY ARGUELLO FUENTES

CODIRECTOR
ING. ERICK RAMON MENÉSES CUADROS

INGENIERÍA DE SISTEMAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA, 2009

Este libro está dedicado a mis padres Esperanza y Jorge por su gran esfuerzo para darme siempre lo mejor,

A Lina por su apoyo incondicional quien ha vivido conmigo los últimos logros alcanzados y con quien espero compartir muchos más,

A mis hermanos que a pesar de la distancia siempre he sentido su calor y apoyo en todo lo que me he propuesto.

Luis Eduardo

Dedico este libro

A mis padres Rosa y Victor cuyo afecto y comprensión han sido mi inspiración

A Nancy Rocio porque en su compañía la tristeza se transforma en alegría y la soledad no existe.

Mario Julián

Agradezco a mis padres y hermanos por que siempre han creído en mí,

A Cusol el cual considero un espacio de diversidad de ideas y de acciones altruistas,

Al Ingeniero Erick Menéses por su gran amistad y por la orientación académica brindada,

A Mario Julián quien ha sido un gran amigo, por su incondicional apoyo y por su leal compañía en muchos de los logros obtenidos hasta ahora,

A mis amigos de siempre con quienes compartí momentos buenos y malos, aquellos con los que aprendí el valor de la amistad,

A mi tía Doris y a toda su familia quienes han sido un gran soporte en este proceso,

A todas las personas que conocí en este claustro universitario quienes aportaron en mi realización profesional.

Luis Eduardo

Agradecer hoy y siempre a mis padres por su afecto y apoyo incondicional.

A Luis Eduardo por todo el ánimo, toda la paciencia, por confiar en mí, por ser como una hermano y sobre todo por su valiosa amistad.

A Erick Meneses por brindarme su apoyo, colaboración y por los momentos en los que más que un profesor se comportó como un amigo.

A la Universidad Industrial de Santander y a los profesores de la Escuela de Ingeniería de Sistemas por su apoyo académico e investigativo.

Al CENTIC por el apoyo de sus recursos tecnológicos.

A CUSOL espacio de ideas nuevas y diferentes.

Y a todos los amigos que de una u otra forma, colaboraron o participaron en la realización de esta investigación.

Mario Julián

Tabla de contenido

Índice de Ilustraciones

Índice de Tablas

Índice de Anexos

GLOSARIO

Bit de paridad: Consiste en añadir a los n bits que forman el carácter original, un bit extra, llamado bit de paridad. Este bit se elige por el número total de bits "1", si el número es par el código de paridad par es 0 o impar el código de paridad impares 1.

Downtime: (tiempo de inactividad del sistema), Es usado para definir cuando el sistema no está disponible (fuera de servicio) debido a reparaciones/mantenimiento o averías.

Failback: Es el proceso de devolución de uno o más servicios a su nodo principal (después de que el nodo principal esté disponible de nuevo). En otras palabras, failback es el proceso inverso de failover.

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. Una situación de failover es una situación excepcional, para la cual ha sido concebida la alta disponibilidad, el fallo de un nodo.

GNU/Linux: Es la denominación defendida por Richard Stallman para el sistema operativo que utiliza el Kernel Linux en conjunto con las aplicaciones de sistema creadas por el proyecto GNU[1], que conforman en conjunto un sistema operativo libre.

HTTP: *P*rotocolo de transferencia de hipertexto (HTTP, HyperText Transfer Protocol) es el protocolo usado en cada transacción de la Web. Define la sintaxis y la semántica que utilizan los elementos software de la arquitectura Web (clientes, servidores, proxys) para comunicarse.

Scheduler: Es el componente que por medio de una política administra el conjunto de recursos y los servicios o procesos que necesiten de este, de manera que intenta satisfacer los objetivos de maximizar la utilización de los procesadores y minimizar los costos de la comunicación.

SPOF: (Single Point Of Failure ó punto simple de fallo) Hace referencia a cualquier

[1] Proyecto GNU http://www.gnu.org/

elemento no replicado y que puede estar sujeto a fallos, afectando con ello al servicio.

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

VIP: Dirección IP virtual, por donde los clientes realizan la petición de un servicio al cluster de alta disponibilidad.

RESÚMEN

Titulo: DISEÑO E IMPLEMENTACIÓN DE UNA INFRAESTRUCTURA PARA LA PRESTACIÓN DEL SERVICIO DE CORREO USANDO LA TÉCNICA DE CLUSTERING DE ALTA DISPONIBILIDAD. [*]

Autores: BONILLA CONTRERAS, Mario Julián [**]
VARGAS BECERA, Luis Eduardo

Palabras clave: clúster de computadores, servicios informáticos, sistemas distribuidos, balanceo de carga, monitoreo de servicios, tolerancia a fallos, escalabilidad.

Descripción: En la actualidad las organizaciones requieren de servicios informáticos 7/24 para operar correctamente, servicios que estén disponibles 7 días a la semana 24 horas al día. Para muchas organizaciones, las interrupciones no planeadas representan tiempo sin el apoyo de sus sistemas de informáticos y puede llegar a ser catastrófico, o al menos muy costoso. Con el fin de aumentar los niveles de disponibilidad para ganar ventaja competitiva en las organizaciones, se ha llegado al surgimiento de alternativas tecnológicas revolucionarias como son los cluster de alta disponibilidad.

En respuesta a estas necesidades el presente proyecto propone el uso de la técnica de clustering de alta disponibilidad para la prestación del servicio de correo en organizaciones con gran número de usuarios y alta demanda del servicio, para lo cual se realiza el diseño e implementación de una arquitectura distribuida que contempla aspectos como el balanceo de carga (algoritmos de planificación y monitoreo de servicios), el almacenamiento masivo (sistemas de archivos distribuidos con alto rendimiento y tolerancia a fallos), y los componentes inherentes al servicio de correo electrónico.

Este esquema usa como técnica principal la redundancia, instalando varios servidores completos (con Pcs normales de escritorio) en lugar de uno sólo como en las arquitecturas centralizadas tradicionales donde se pueden presentar daños o errores internos que afectan el funcionamiento del sistema informático, convirtiéndose en un punto único de fallo (SPOF) del cual depende el funcionamiento del servicio.

Un aspecto importante en este tipo de arquitecturas es la escalabilidad, con la cual se puede lograr incrementar la capacidad de almacenamiento y de mantener o aumentar el nivel de servicio agregando más nodos al clúster, dando la posibilidad de responder de manera flexible ante cambios en la demanda del servicio o almacenamiento.

---

[*] Trabajo de grado

[**] Facultad de Ingenierías Físico-Mecánicas, Ingeniería de Sistemas. Director: Henry Arguello Fuentes.

ABSTRACT

Title: DESIGN AND IMPLEMENTATION OF AN INFRASTRUCTURE FOR THE PROVISION OF MAIL SERVICE USING THE TECHNIQUE OF CLUSTERING FOR HIGH AVAILABILITY.[*]

Autores: BONILLA CONTRERAS, Mario Julián [**]
VARGAS BECERA, Luis Eduardo

Keywords: cluster of computers, computer services, distributed systems, load balancing, monitoring services, fault tolerance, scalability.

Description: Currently, organizations require services 7 / 24 to operate correctly, services that are available 7 days a week 24 hours a day. For many organizations, the interruptions are unplanned time without the support of its computer systems and may become catastrophic, or at least very costly. To increase the level of availability to gain competitive advantage in organizations, it has been the emergence of revolutionary technological alternatives such as high availability clustering.

In response to these needs, this project proposes the use of the technique of clustering for high availability for the provision of mail service in organizations with large numbers of users and high-demand service, which makes the design and implementation of a distributed architecture that includes aspects such as load balancing (algorithms for planning and monitoring services), mass storage (distributed file system with high performance and fault tolerance), and components inherent in the e-mail service.

This scheme uses as the main technical the redundancy by installing multiple servers (with standard desktop PCs) instead of just one as in the traditional centralized architectures which can damage or internal errors that affect the operation of the system, becoming a single point of failure (SPOF) which depends on the operation of the service.

An important aspect in this type of architecture is scalability, with this it's possible to achieve increase storage capacity and maintain or increase the level of service by adding more nodes to the cluster, giving the ability to respond flexibly to changes in the demand for service or storage.

---

[*] Grade Work

[**] Faculty of Physics – Mechanics Engineering, Engineering of Systems and Informatics. Director: Henry Arguello Fuentes.

1. INTRODUCCIÓN

La computación desde su origen, ha servido como herramienta de apoyo en las labores diarias del hombre, especialmente en sus necesidades de comunicación bien sea de manera individual u organizacional, dichas necesidades han hecho de los sistemas de comunicación entre computadoras un elemento de amplia y creciente implantación, sin embargo, con el crecimiento exponencial de la Internet se ha generado una revolución que exige sistemas de cooperación entre organizaciones capaces de administrar e integrar gran cantidad de servicios e información, esta ultima vista como elemento vital que habilita la toma de decisiones acertadas y la entrega de mejores productos y servicios a través del uso de tecnologías de información.

En la actualidad, existen servicios que necesitan ser prestados de manera ininterrumpida los 365 días del año, lo que requiere una mejor organización de los componentes que prestan dichos servicios, dejando de lado la idea de los sistemas centralizados como la única opción de diseño y apostando al uso de sistemas distribuidos que permiten lograr unos niveles de disponibilidad y escalabilidad óptimos según las necesidades organizacionales, además de brindar transparencia y fiabilidad de servicio al usuario final.

En este sentido, surgen tecnologías como la alta disponibilidad (HA)[1] que permite obtener sistemas distribuidos tolerantes a fallos, bajo el uso de la técnica maestra: la redundancia; esta estrategia consiste en replicar las zonas críticas del sistema, teniendo unidades activas y varias copias de respaldo que, tras el fallo de la principal, sean capaces de retomar su labor en el punto que aquella falló (failover), en el menor tiempo posible y de forma transparente para el usuario.

La disponibilidad[3] obtenida con el desarrollo de dicha tecnología se puede medir a través de un índice (un porcentaje) que se obtiene dividiendo el tiempo durante el cual el servicio está disponible por el tiempo total de servicio, así, un buen diseño de un sistema informático para la prestación de un servicio puede alcanzar valores de este índice de 99%, representando en un periodo de un año un total de

3.65 días de no disponibilidad del servicio (Downtime). Una medida más alta de este índice implica decisiones en las organizaciones basadas en la necesidad de tener servicios más disponibles y fiables que conllevan un aumento del costo total del sistema para la prestación de un servicio.

Las organizaciones necesitan soluciones que sirvan para aumentar los niveles de disponibilidad con el fin de ganar ventaja competitiva, es por esto que se ha llegado al surgimiento de alternativas especializadas, equipos con altas prestaciones para multiprocesamiento y redundancia, que dan solución al problema, pero que muchas veces implican la realización de grandes inversiones, con el inconveniente que cuando estos equipos quedan obsoletos no hay mas opción que comprar uno nuevo.

Como respuesta a la necesidad de soluciones para estos servicios que no demanden tanto costo y que permitan un rendimiento igual o superior al de las soluciones obtenidas por medio de equipos especializados, surge *Linux HA Project*[2] que es el encargado de aunar los esfuerzos de la comunidad de software libre para hacer de GNU/Linux una excelente plataforma sobre la cual ofrecer servicios de HA y escalabilidad.

El presente proyecto tiene por objetivo implementar un cluster en alta disponibilidad para un servicio de correo a partir de una arquitectura diseñada para tal fin, en donde se consideren aspectos como el almacenamiento, el balanceo de carga y los componentes inherentes al correo electrónico. Este esquema usa como técnica principal la redundancia, instalando varios servidores completos (con Pcs normales de escritorio) en lugar de uno sólo, que sean capaces de trabajar coordinadamente y de asumir las caídas de algunos de los nodos que lo componen, además se puede agregar y remover nodos del sistema según las necesidades (escalabilidad).

En los capítulos siguientes se hace una descripción del problema planteado en este proyecto, la justificación de su desarrollo, la descripción de las técnicas y herramientas utilizadas para la implementación del cluster de alta disponibilidad,

[2] Linux-HA es un proyecto ampliamente utilizado en muchas soluciones de alta disponibilidad, http://www.linux-ha.com/.

los test realizados para la elección del sistema de archivos a utilizar, y las pruebas de funcionamiento del servicio para una comunidad de usuarios en el Centro de Tecnologías y Comunicaciones CENTIC.

2. ESPECIFICACIONES DEL PROYECTO

2.1 ESPECIFICACIONES DEL PROYECTO

2.2 TÍTULO

DISEÑO E IMPLEMENTACIÓN DE UNA INFRAESTRUCTURA PARA LA PRESTACIÓN DEL SERVICIO DE CORREO USANDO LA TÉCNICA DE CLUSTERING DE ALTA DISPONIBILIDAD.

2.3 DIRECTOR

MPE. Henry Arguello Fuentes.

Universidad Industrial de Santander, Bucaramanga Colombia.

2.4 CODIRECTOR

Ing. Erick Ramón Meneses Cuadros.

Universidad Industrial de Santander, Bucaramanga Colombia.

erick@linux.com.co

2.5 AUTORES

Mario Julián Bonilla Contreras. Luis Eduardo Vargas Becerra

Estudiante de Ingeniería de Sistemas Estudiante de Ingeniería de Sistemas

julian.1221@gmail.com luivar22@gmail.com

3. JUSTIFICACIÓN

En la actualidad las organizaciones requieren de servicios informáticos 7/24 para funcionar, servicios que estén disponibles 7 días a la semana 24 horas al día. Para muchas empresas, las interrupciones no planeadas representan tiempo sin el apoyo de sus sistemas de informáticos y puede llegar a ser catastrófico, o al menos muy costoso, como lo revelan dos estudios independientes realizados por Giga Group[3] y Eagle Rock Alliance,Ltd[4], encontrando que una empresa media pierde hasta 6.45 millones dólares por hora de interrupción no planeada de sus servicios informáticos.

En organizaciones en continuo crecimiento la demanda esperada del servicio de correo electrónico en ocasiones es superada, bajando el nivel del servicio y dejando ver la falta de disponibilidad, la cual es interrumpida por eventos internos, tales como errores de procesadores, errores de software, daños de discos, caídas de comunicaciones, errores de operación, vandalismo o sabotaje, que plantean un peligro real en tiempos muertos no planificados de los servicios.

Aquí conviene detenerse un momento porque en este contexto, surge como alternativa la técnica de clustering de alta disponibilidad para solucionar los requerimientos de este tipo de servicios. Un cluster de alta disponibilidad está diseñado para aumentar el funcionamiento ininterrumpido de ciertas aplicaciones, mediante la replicación de datos y redundancia de recursos hardware y software.

En respuesta a estas necesidades el presente proyecto de grado da como resultado el diseño e implementación de una infraestructura de computo distribuido para lograr un servicio de correo electrónico en continua disponibilidad, escalable y con un sistema de archivos distribuido tolerante a fallos basado en la técnica de clustering de alta disponibilidad y haciendo uso de los recursos

[3] Firma de analistas internacional en tecnología de información. Resultados del estudio se encuentran disponibles en: http://www.dba-oracle.com/art_dbazine_high_avail.htm.

[4] Eagle Rock Alliance, Ltd es una empresa consultora especializada en administración de negocios, en la planificación de Fiabilidad de consultoría y tecnología. Resultados del estudio se encuentran disponibles en: http://www.beechtek.net/page/1012471

computacionales del Centro de Tecnologías de la Información y Comunicaciones (CENTIC), además es en la comunidad CENTIC (personal administrativo y desarrolladores de proyectos) que el cluster funcionará prestando el servicio de correo electrónico.

Esta infraestructura de computo es un prototipo que dará como resultado un cluster funcional de forma experimental, sin embargo, es de nombrarse que una vez terminado el trabajo podría extenderse la propuesta a otras instituciones y empresas de manera más amplia brindando soluciones a cualquier servicio que exija una alta disponibilidad.

4. DESCRIPCIÓN DEL PROBLEMA

Internet es un medio de comunicación importante y popular en todo el mundo y con la necesidad de buscar formas de compartir información nace el correo electrónico, que comenzó a utilizarse en 1965 en una supercomputadora de tiempo compartido y, para 1966, se había extendido rápidamente para utilizarse en las redes de computadoras.

El e-mail (electronic mail) es un método de comunicación entre dos o más personas, que de manera sencilla, casi instantánea, sin limitaciones espaciales, permite la transmisión de información, pudiendo agregarse cualquier tipo de esta, entre ellos: texto, fotos, sonidos, etc. Este sistema de acceso inmediato y directo entre los seres humanos ha revolucionado el mercado de la comunicación desde hace un tiempo masificando progresivamente su uso, llevando al incremento de la demanda esperada en dicho servicio, el cual no puede bajar su nivel de prestaciones en cuanto a la disponibilidad y capacidad para mantenerse operativo (fiabilidad), razones de peso para el surgimiento de nuevas tecnologías y procesos que mejoren su eficiencia y calidad dadas sus características dinámicas de crecimiento[4]. Esta masificación y crecimiento de la demanda se debe a las facilidades tecnológicas actuales ofrecidas por redes como Internet, que permiten la inclusión de cada ves mas usuarios u organizaciones motivados por la tecnología, necesidad o curiosidad llegan a reemplazar los métodos tradicionales de comunicación por e-mail, aprovechando todas sus ventajas y facilidades, poniendo en evidencia que servicios como el de correo electrónico no puede estar sujeto a un número de usuarios limitado, y que debe estar en la capacidad de ser escalable cuando este se incremente.

En las arquitecturas centralizadas tradicionales para la prestación de servicios como el de correo electrónico se pueden presentar daños o errores internos que afectan el funcionamiento del sistema informático, convirtiéndose en un punto único de fallo (SPOF)[5] del cual depende el funcionamiento del servicio, es clara la necesidad de la utilización de la computación distribuida y redundancia de recursos para evitar los puntos únicos de fallo, por lo que se hace indispensable

generar una arquitectura computacional capaz de brindar un servicio ininterrumpido, un esquema teórico con soporte tecnológico y validado a través de pruebas, lo cual se constituye en el objetivo de este proyecto.

5. OBJETIVOS

5.1 OBJETIVO GENERAL

Diseñar e implementar un servidor de correo, por medio de una infraestructura distribuida altamente disponible para incrementar el funcionamiento ininterrumpido tanto en servicio y la integridad en almacenamiento, haciendo uso de un sistema de archivos distribuido tolerante a fallos.

5.2 OBJETIVOS ESPECÍFICOS

- Diseñar una arquitectura que contemple la integración de componentes hardware y software para prestar el servicio de correo electrónico basado en cluster de alta disponibilidad [2], teniendo en cuenta aspectos como: el balanceo de carga, disponibilidad en la prestación del servicio y el tratamiento distribuido de los datos.
- Comparar teóricamente y seleccionar a partir de una revisión bibliográfica, las herramientas tecnológicas (balanceo de carga, componentes inherentes al servicio de correo y sistema de archivos distribuido [9]) que en conjunto brinden las condiciones adecuadas para la implementación del diseño del servicio de correo electrónico.
- Implementar la infraestructura tecnológica del servicio de correo, a partir del diseño planteado y, las herramientas y protocolos seleccionados, que instalados, configurados y acoplados correctamente, permitirán generar una solución eficiente al problema.
- Realizar pruebas de funcionalidad en la arquitectura implementada mediante la creación de usuarios (trabajadores y administrativos del Centic) del servicio de correo y envío de mensajes a nivel local y remoto (usando protocolos como SMTP, IMAP y POP), para escenarios diseñados donde se presenten eventos de failover y failback que muestren la disponibilidad del servicio.

6. MARCO TEÓRICO

6.1 ALTA DISPONIBILIDAD

El término disponibilidad se refiere a la cantidad de tiempo que un sistema está proporcionando servicio, en relación al tiempo en que los usuarios desean utilizarlo. La disponibilidad [8] suele cuantificarse con la siguiente ecuación:

$$D = \left(\frac{MTTF}{MTTF + MTTR} \right) * 100 \quad (1)$$

Donde MTTF[5] es el tiempo medio durante el cual el sistema está proporcionando servicio y MTTR[6] es el tiempo medio que tarda el sistema en ser reparado o acondicionado para su funcionamiento. La disponibilidad suele expresarse en tanto por ciento, y permite clasificar a los sistemas en función del grado de disponibilidad que proporcionan. Así, a los sistemas disponibles menos del 99.99% del tiempo se los suele clasificar como sistemas de disponibilidad baja o media, mientras que se consideran altamente disponibles a aquellos que superen el 99.99% de disponibilidad (una indisponibilidad de menos de 1 hora al año aproximadamente). En este rango se suelen situar las pretensiones de los clusters, que demandando cierto valor de disponibilidad, asumen como tolerable ciertas indisponibilidades temporales de reducida duración (del orden de segundos las más prolongadas). En el escalafón más exigente se encuentran las aplicaciones que demandan ultra-alta disponibilidad o disponibilidad total, como por ejemplo aplicaciones de control de centrales nucleares, de navegación automatizada o de mantenimiento de constantes vitales de seres humanos.

6.2 COMPUTACIÓN DISTRIBUIDA

El uso de varios computadores interconectados por redes de comunicaciones dio

[5] Del inglés Mean Time To Failure, o tiempo medio hasta fallo.

[6] Del inglés Mean Time to Repair, o tiempo medio para reparar.

lugar a la aparición hace varias décadas del concepto de sistema distribuido. Los sistemas distribuidos son una agrupación de recursos que trabajan de forma conjunta para realizar una tarea. Esta definición es similar a las que se puede encontrar en multitud de fuentes [6][7].

6.2.1 COMPONENTES DE UN SISTEMA DISTRIBUIDO

La definición de sistemas distribuidos queda más completa si se enumeran los tres componentes [6] que forman:

- *Computadores*: todo sistema distribuido está formado por más de un computador físico, al que también se le denomina nodo del sistema distribuido o simplemente nodo. Los nodos al menos tendrán CPU, memoria y dispositivos de entrada/salida que les permitan comunicarse con el entorno.
- *Red*: interfaces de comunicaciones que permiten interconectar a los nodos del sistema distribuido. Elemento fundamental a la hora del diseño del sistema, ya que se debe tener en cuenta que en los sistemas centralizados los canales de comunicación entre los procesos tienen menor latencia que las redes convencionales.
- *Estado compartid*o: los nodos de un sistema distribuido mantienen un estado compartido y para mantener este estado de forma correcta, deberán coordinarse entre ellos. Esta característica marca la diferencia fundamental entre los sistemas distribuidos y los centralizados donde estado del sistema está en un único componente.

6.2.2 CARACTERÍSTICAS DE LOS SISTEMAS DISTRIBUIDOS

Una de las características más importantes de un sistema distribuido, es su capacidad para tolerar fallos en algunos de sus componentes. A diferencia de lo

que ocurre en un sistema centralizado, en un sistema distribuido habrá múltiples unidades de cómputo y de gestión de recursos que serán independientes (en cuanto a su probabilidad de fallo). Por tanto, la probabilidad de que falle el sistema en su totalidad será menor. No obstante, para garantizar un buen funcionamiento en un servicio de información a través de un sistema distribuido, es necesario emplear técnicas de alta disponibilidad, como la replicación de componentes, que permitirán que el sistema siga comportándose de acuerdo con sus especificaciones incluso cuando alguno de sus elementos falle.

Otra ventaja de estos sistemas radica en su escalabilidad, es decir, en la posibilidad de aumentar la capacidad de servicio que se obtiene al agregar nuevos nodos y conectarlos al sistema que ya existía, reconfigurándolo para que estas nuevas unidades pasen a ser también utilizadas. Al agregar más unidades al sistema, se adquiere más capacidad de procesamiento y de cálculo pudiéndose dar solución a problemas de computo más rápidamente además de poder solucionar problemas mas extensos y con mayor complejidad.

6.3 CLUSTERS COMPUTACIONALES

El término *cluster* se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de componentes de hardware que no son necesariamente comunes, y que se comportan como si fuesen una única computadora. La tecnología de clusters ha evolucionado en apoyo de actividades que van desde aplicaciones de supercómputo y software de misiones críticas, servidores Web y comercio electrónico, hasta bases de datos de alto rendimiento, entre otros usos. Por lo tanto el cómputo con clusters surge como resultado de la convergencia de varias tendencias actuales que incluyen la disponibilidad de microprocesadores económicos de alto rendimiento y redes de alta velocidad, el desarrollo de herramientas de software para cómputo distribuido de alto rendimiento, así como la creciente necesidad de potencia computacional para aplicaciones que la requieran.

6.3.1 TIPOS DE CLUSTER

6.3.1.1 CLUSTER ALTO RENDIMIENTO

Es un conjunto de ordenadores que está diseñado para dar altas prestaciones en cuanto a capacidad de cálculo.

6.3.1.2 CLUSTER DE ALTA DISPONIBILIDAD

Es un conjunto de dos o más máquinas que se caracterizan por mantener una serie de servicios compartidos y por estar constantemente monitorizándose entre sí, de tal forma que cuando una de las máquinas falla, la otra toma su lugar en los servicios que la primera estaba prestando, brindando de manera transparente al usuario integridad en la información y fiabilidad en el servicio, quien no notará que hubo un fallo en el sistema.

6.3.1.3 CLUSTER DE BALANCEO DE CARGA

Conjunto de dos o más máquinas que actúan como entrada de un cluster, y que se ocupan de repartir con algún algoritmo de planificación peticiones de servicio recibidas a otras máquinas para ser procesadas posteriormente. Tiene como características la escalabilidad al permitir a agregar más máquinas al cluster.

6.4 CLUSTER DE ALTA DISPONIBILIDAD

Un Cluster de Alta disponibilidad (Cluster HA) es un tipo de cluster computacional que se realiza fundamentalmente con el fin de mejorar la disponibilidad de servicios en una red (servidores Web, de correo, de bases de datos, entre otros). Estos operan teniendo redundancia elementos o nodos que se utilizan para prestar el servicio aún cuando fallen algunos de los componentes del sistema, dado que se quiere evitar lo ocurrido en los sistemas centralizados donde cuando una aplicación falla, no estará disponible hasta que alguien haga una reparación

del servidor. En vista de estas situaciones, los clusters HA dan solución a esta problemática mediante la detección de defectos de hardware y software, e inmediatamente se vuelve a arrancar la aplicación en otro sistema sin necesidad de intervención administrativa.

Como parte de este proceso, el software del cluster puede configurar el nodo antes de iniciar la aplicación en él, implicando ajustes en el sistema tales como: sistemas de archivos que necesiten ser importados y montados, configuraciones en el hardware de red de los nodos, y algunas aplicaciones de soporte que necesiten correrse también. Por lo tanto, el software existente en estos sistemas debe garantizar todas estas configuraciones y debe poder soportar situaciones dinámicas de desempeño en caso de fallos en algunos de los componentes del sistema (hardware y software). Para hacer frente a situaciones de fallos se hace uso de la replicación de máquinas y dispositivos cuya finalidad es evitar los puntos únicos de fallo, del inglés SPOF (Single Point of Failure), que serían aquellas máquinas y elementos (conexiones de red, dispositivos de almacenamiento, entre otros) imprescindibles para el correcto funcionamiento del servicio que quiere dar.

En consecuencia, han surgido esfuerzos para dar solución a tales situaciones por medio del uso de herramientas software de uso libre y código abierto, que se han unido en dos proyectos: el proyecto Linux-HA[7], y el proyecto LVS[8] (Linux Virtual Server). El primero brinda diversas soluciones en alta disponibilidad para diferentes servicios como Web, correo electrónico, bases de datos, entre otros. El segundo toma los beneficios del primero y aporta una solución de código abierto para gestionar el balanceo de carga para en un cluster de servidores reales añadiendo mas niveles y capas, elementos fundamentales para el diseño de sistemas distribuidos basados en clusters de alta disponibilidad.

Así, los niveles que se contemplan en el proyecto *Linux Virtual Server* son: balanceo de carga (asignación de carga de trabajo sobre los nodos del cluster), monitorización de servicios y aplicaciones (verificación constante de disponibilidad de los nodos y sus servicios), compatibilidad con sistemas de archivos

[7] Proyecto Linux-HA http://www.linux-ha.com/
[8] Proyecto Linux Virtual Server http://www.linuxvirtualserver.org

(distribuidos, paralelos y replicación de la información) en cuanto al almacenamiento y compatibilidad con los servicios a prestar (Web, mail, ftp, multimedia, bases de datos, etc.).

6.4.1 ARQUITECTURA DEL PROYECTO LVS PARA CLUSTERS DE ALTA DISPONIBILIDAD

El proyecto LVS (Linux Virtual Server) provee la información y todos las herramientas software necesarias para montar un "servidor virtual"[9] fácilmente escalable sobre un cluster de máquinas reales con sistema operativo GNU/Linux.

El servidor estará encapsulado para los usuarios finales, quienes pensarán en una sola máquina que servirá sus peticiones. Internamente existirá una arquitectura de cluster de PC's para servir dichas peticiones sobre la cual se distribuirá la carga, lo que resulta en un aumento de la capacidad de atención y procesamiento de solicitudes, como también en un incremento del tiempo disponible de servicio dado que no se depende de una sola máquina que pueda fallar en cualquier momento (existencia de SPOF).

Dentro de la arquitectura de cluster planteada por el proyecto Linux Virtual Server que se denomina "Servidor Virtual", se encuentra el sistema clave que permite la distribución de carga en el cluster, el cual es el sistema balanceador de carga (*Load Balancer*).

El balanceador de carga es un sistema que se ubica entre los clientes y los servidores (ilustración 1), que centraliza la recepción de peticiones. Este sistema está compuesto por un grupo de PC's que están en continuo monitoreo de sus miembros y que tienen la tarea de reenviar las peticiones a los servidores reales, que son los encargados de procesarlas. El balanceador de carga toma las decisiones de reenvío de peticiones en función de un algoritmo de *scheduling* determinado. El método de reenvío puede implementarse de varias formas, en función de la arquitectura de red disponible para comunicar a los servidores reales

[9] Servidor Virtual http://www.linuxvirtualserver.org/whatis.html

con el balanceador de carga. El proyecto Linux-HA provee de herramientas para la monitorización del sistema balanceador, que en este caso está constituido por dos computadores, los cuales estarán en continua comunicación compartiendo un estado (Activo/Pasivo), lo que significa que uno estará como principal, recibiendo peticiones del servicio de correo mientras el otro estará en estado de espera para retomar las peticiones en caso de un fallo en el principal.

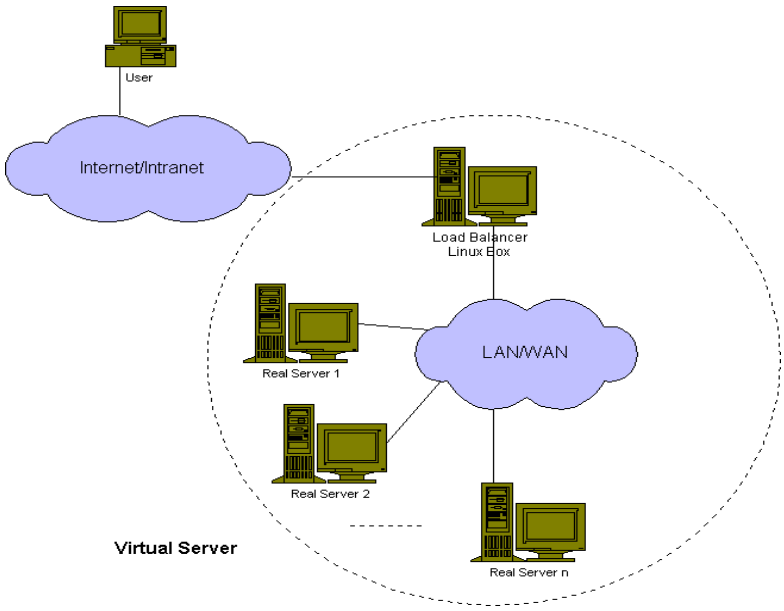


Ilustración 1: Virtual Server

Autor: http://www.linux.virtualserver.org/whatis.html

En los siguientes capítulos se hace una descripción de las bases teóricas y tecnológicas necesarias para entender el diseño de la arquitectura implementada basada en el modelo del proyecto LVS, deteniéndose en cada nivel para enfatizar la importancia y explicación de su funcionamiento.

6.5 MONITORIZACIÓN DE SERVICIOS Y BALANCEO DE CARGA

6.5.1 HEARTBEAT

Una de las herramientas software provistas por el proyecto Linux HA es *heartbeat*

cuya labor es mantener servicios altamente disponibles. Este basa su funcionamiento en el envío y recepción de “latidos”, que son las señales enviadas por los demonios heartbeat que corren en las máquinas del sistema balanceador. Heartbeat es el encargado de iniciar y de detener el servicio de monitoreo a través de la misma IP en todas las máquinas del sistema balanceador. Puesto que no pueden existir dos direcciones IP iguales en el mismo segmento de red, se utiliza una dirección IP virtual (VIP). Heartbeat en el encargado de lanzar la recepción del servicio que ha de prestarse en alta disponibilidad y activa la VIP, cuando detecta la ausencia de latidos.

6.5.2 BALANCEO DE CARGA

El proceso de balanceo de carga hace uso de la lista de servidores reales disponibles que se obtienen mediante una herramienta de administración llamada Ldirectord provista por el proyecto LVS, la cual en tiempo real asigna unos recursos a una solicitud de servicio dependiendo del algoritmo de planificación que se elija.

6.5.2.1 ADMINISTRACIÓN DE LOS SERVIDORES REALES CON LDIRECTORD

Ldirectord (Linux Director) es un demonio lanzado por heartbeat para la monitorización y control de los servidores reales, así como para llevar a cabo la administración de las tablas de forwarding del kernel. Ldirectord prueba el estado de los servidores reales en intervalos de tiempo predeterminados con el objetivo de mantener actualizada la tabla del servidor virtual haciendo uso de la interfaz de usuario ipvsadm (administrador del balanceador de carga IPVS).

De esta manera, ldirectord permite actualizar dinámicamente las tablas que definen el comportamiento del balanceador de carga IPVS. Si uno de los servidores reales deja de servir peticiones de alguno de los servicios prestados

por el cluster de alta disponibilidad, ldirectord creará una llamada a ipvsadm que hará que el balanceador deje de reenviarle paquetes al servidor real.

6.5.2.2 ALGORITMOS DE PLANIFICACIÓN

El sistema balanceador es el encargado de distribuir la carga entre los diferentes servidores reales. Para hacerlo se basa en un conjunto secuencial de normas, llamadas algoritmos de planificación (scheduling), que permiten decidir en cada momento a que servidor debe reenviar los paquetes pertenecientes a una misma conexión. Dentro de los algoritmos que se pueden usar con el proyecto LVS[1] se encuentran los listados posteriormente:

- *Round Robin (RR)*, algoritmo donde cada petición se envía a un servidor real, y la siguiente petición al siguiente servidor real de la lista, hasta llegar al último tras lo cual se vuelve a enviar al primero. Es la solución más sencilla y que menos recursos consume, a pesar de que no es la más justa, ya que es posible que toda la carga "pesada" sea atendida por el mismo servidor real mientras que el resto sólo reciban peticiones triviales. Otro problema de este método es que todos los servidores recibirán el mismo número de peticiones, independientemente de si su potencia de cálculo es la misma o no.
- *Round Robin Ponderado (WRR)*, este algoritmo es igual que el anterior, pero añadiendo un "peso" a cada servidor. Este peso es un entero que indica la potencia de cálculo del servidor, de forma que la cola Round Robin se modificará para que aquellos servidores con mayor potencia de cálculo reciban peticiones más a menudo que el resto. Por ejemplo, si tienen tres servidores reales A, B y C, con una cola Round Robin normal la secuencia de distribución tendrá tres pasos y será ABC. Si se usa una Round Robin Ponderada y asigna pesos con valor de 4, 3 y 2 respectivamente a cada servidor, la cola ahora distribuirá en nueve pasos (4+3+2) y una posible planificación de acuerdo a estos pesos sería AABABCABC.

- *Servidor con menos conexiones activas* (LC), este mecanismo de distribución consulta a los servidores reales para ver en cada momento cuántas conexiones abiertas tiene cada uno con los clientes, y envía cada petición al servidor que menos conexiones tenga en ese momento. Es una forma de distribuir las peticiones hacia los servidores con menos carga.
- *Servidor con menos conexiones activas ponderado (WLC)*, al igual que la estrategia Round Robin Ponderada, en este algoritmo se toma el anterior y se le añaden unos pesos a los servidores reales que de alguna forma midan su capacidad de cálculo, para modificar la preferencia a la hora de escoger uno u otro según este peso.
- *Servidor con menos conexiones basándose en servicio*, este algoritmo dirige todas las peticiones a un mismo servidor, hasta que se sobrecarga (su número de conexiones activas es mayor que su peso) y entonces pasa a una estrategia de menos conexiones activas ponderada sobre el resto de servidores del cluster. Este método de planificación puede ser útil cuando se ofrecen varios servicios distintos y quiere especializar cada máquina en un servicio, pero siendo todas ellas capaces de reemplazar a las demás.
- *Tablas hash por origen y destino,* en este método se dispone de una tabla de asignaciones fijas, en las que bien por la IP de origen o de destino, se indica qué servidor deberá atender la petición. El sistema balanceador compara las direcciones de las tramas TCP/IP que reciba con estas tablas y actúa en consecuencia.

6.6 SERVICIO DE CORREO

En este capítulo se hace una descripción de funcionamiento de un servicio de correo, los protocolos usados y la descripción de sus componentes.

6.6.1 PROTOCOLOS Y EXTENSIONES PARA EL SERVICIO DE CORREO ELECTRÓNICO

6.6.1.1 SMTP

(Simple Mail Transfer Protocol, en español Protocolo Simple de Transferencia de Correo) Protocolo de red basado en texto utilizado para el intercambio de mensajes de correo electrónico entre computadoras o distintos dispositivos (PDA's, teléfonos móviles, etc.) el cual es estándar en Internet.

SMTP se basa en el modelo cliente-servidor, donde un cliente envía un mensaje a uno o varios receptores. La comunicación entre el cliente y el servidor consiste enteramente en líneas de texto compuestas por caracteres ASCII[10]. Las respuestas del servidor constan de un código numérico de tres dígitos, seguido de un texto explicativo. En el protocolo SMTP todas las órdenes, réplicas o datos son líneas de texto, delimitadas por el carácter <CRLF>. Todas las réplicas tienen un código numérico al comienzo de la línea.

En el conjunto de protocolos TCP/IP, el SMTP va por encima del TCP, usando normalmente el puerto 25 en el servidor para establecer la conexión.

6.6.1.2 MIME

(Multipurpose Internet Mail Extensions), (Extensiones de Correo de Internet Multipropósito), son una serie de convenciones o especificaciones dirigidas a que se puedan intercambiar a través de Internet todo tipo de archivos (texto, audio, vídeo, etc.) de forma transparente para el usuario. Una parte importante del MIME está dedicada a mejorar las posibilidades de transferencia de texto en distintos idiomas y alfabetos. En sentido general las extensiones de MIME van encaminadas a soportar:

- Adjuntos que no son de tipo texto.
- Cuerpos de mensajes con múltiples partes (multi-part).

[10] http://es.wikipedia.org/wiki/ASCII

- Información de encabezados con conjuntos de caracteres distintos de ASCII.

Prácticamente todos los mensajes de correo electrónico escritos por personas en Internet y una proporción considerable de estos mensajes generados automáticamente son transmitidos en formato MIME a través de SMTP. Los mensajes de correo electrónico en Internet están cercanamente asociados con el SMTP y MIME por lo que usualmente se les llama mensaje SMTP/MIME.

6.6.1.3 ESMTP

El servicio ESMTP (por sus siglas en inglés – Enhanced Simple Mail Transfer Protocol), es una definición de extensiones de protocolo para el estándar SMTP. Con este se estableció una estructura para todas las extensiones existentes y futuras con el fin de producir una manera consistente y manejable por la cual los clientes y servidores SMTP puedan ser identificados y los servidores SMTP puedan señalar las extensiones soportadas a los clientes conectados.

La característica de identificación principal para ESMTP es que los clientes abren una transmisión con el comando EHLO (Extended HELLO) en lugar de HELO (el Hello original del estándar de SMTP sencillo. Un servidor puede por tanto responder con éxito (código 250), falla (código 550) o error (códigos 500, 501, 502, 504 o 421), dependiendo de su configuración. Un servidor ESMTP respondería el código 250 OK en una respuesta de varias líneas con su dominio y una lista de palabras clave para indicar las extensiones soportadas, en cambio un servidor sumiso SMTP retornaría el código de error 500, permitiendo al cliente ESMTP intentar tanto HELO como QUIT.

6.6.1.4 IMAP

(Internet Message Access Protocol) Protocolo de red de acceso a mensajes electrónicos almacenados en un servidor. Mediante IMAP se puede tener acceso al correo electrónico desde cualquier equipo que tenga una conexión a Internet.

IMAP tiene varias ventajas sobre POP (Post Office Protocol) que es el otro protocolo empleado para obtener correo desde un servidor entre las que se encuentran la complejidad ya que IMAP permite visualizar los mensajes de manera remota y no descargando los mensajes como lo hace POP. A continuación se describen brevemente las ventajas [22][28] del uso de IMAP sobre POP:

- Modo de operación: al utilizar POP, los clientes se conectan brevemente al servidor de correo, solamente el tiempo que les tome descargar los nuevos mensajes. Al utilizar IMAP, los clientes permanecen conectados el tiempo que su interfaz permanezca activa y descargan los mensajes bajo demanda. Esta manera de trabajar de IMAP puede dar tiempos de respuesta más rápidos para usuarios que tienen una gran cantidad de mensajes o mensajes grandes.
- *Conexión de múltiples clientes simultáneos a un mismo destinatario:* el protocolo POP supone que el cliente conectado es el único dueño de una cuenta de correo. En contraste, el protocolo IMAP permite accesos simultáneos a múltiples clientes y proporciona mecanismos a los clientes para que se detecten los cambios hechos a un mailbox por otro cliente concurrentemente conectado.
- *Acceso a partes MIME de los mensajes:* el protocolo IMAP le permite a los clientes obtener separadamente cualquier parte MIME individual, así como obtener porciones de las partes individuales o los mensajes completos.
- *Información de estado de los mensajes:* A través de la utilización de señales definidas en el protocolo IMAP de los clientes, se puede vigilar el estado del mensaje, por ejemplo, si el mensaje ha sido o no leído, respondido o eliminado. Estas señales se almacenan en el servidor, de manera que varios clientes conectados al mismo correo en diferente tiempo pueden detectar los cambios hechos por otros clientes.
- *Accesos múltiples a los buzones de correo en el servidor:* Los clientes de IMAP pueden crear, renombrar o eliminar correo (por lo general presentado

como carpetas al usuario) del servidor, y mover mensajes entre cuentas de correo. El soporte para múltiples buzones de correo también le permite al servidor proporcionar acceso a los directorios públicos y compartidos.

- *Búsquedas de parte del servidor:* IMAP proporciona un mecanismo para que los clientes pidan al servidor que busque mensajes de acuerdo a una cierta variedad de criterios. Este mecanismo evita que los clientes descarguen todos los mensajes de su buzón de correo, agilizando, de esta manera, las búsquedas.
- *Mecanismo de extensión definido:* IMAP define un mecanismo explícito mediante el cual puede ser extendido. Se han propuesto muchas extensiones de IMAP y son de uso común. Un ejemplo de extensión es el IMAP_IDLE, que sirve para que el servidor avise al cliente cuando ha llegado un nuevo mensaje de correo y éstos se sincronicen. Sin esta extensión, para realizar la misma tarea, el cliente debería contactar periódicamente al servidor para ver si hay mensajes nuevos.

6.6.2 COMPONENTES DE UN SERVIDOR DE CORREO ELECTRÓNICO

Un servidor de correo electrónico es un conjunto de herramientas software que permiten el manejo de protocolos, para el envío y recepción de mensajes en la red. La tarea de enviar y recibir mensajes de correo a través de Internet está dividida en labores que realizan diferentes componentes, entre los que se encuentran: el MTA (Mail Transport Agent*,* en español Agente de Transporte de Correos), MUA (Mail User Agent, en español Agente de Usuario de Correo), MDA (Mail Delivery Agent, en español Agente Repartidor de Correo). A continuación se hace una breve descripción de estos componentes[25]:

- MTA: es el encargado de comunicarse por medio del protocolo SMTP y sus extensiones, con otros MTA o con los MUA. Ejemplo: qmail, sendmail, postfix, exim.

- MUA: programa que permite enviar y leer los mensajes de una cuenta de correo electrónico. Ejemplo: Microsoft Office Outlook, Mozilla Thunderbird, Roundcubemail, Squirrelmail.
- MDA: es un software que acepta correo entrante y los distribuye a los buzones de los destinatarios (si la cuenta de destino está en la máquina local), o lo reenvía a un servidor SMTP (si los destinatarios están en máquinas remotas). Ejemplo: procmail, maildrop, cyrdeliver.

6.6.2.1 WEBMAIL

En la actualidad es común el uso de clientes de correo en Internet, existen empresas privadas que dan servicio de Webmail (clientes de correo electrónico o MUA), que proveen una interfaz Web por la que acceder al correo electrónico. Algunas de las empresas son: Gmail y Hotmail.

El *Webmail* permite listar, desplegar y borrar vía navegador Web los correos almacenados en el servidor remoto por medio del protocolo IMAP. Los correos pueden ser consultados posteriormente desde otro computador conectado a la misma red (por ejemplo Internet) y que disponga de un navegador Web. Algunos Webmail son: RoundCube, SquirrelMail, Zimbra.

6.7 PROCESO DE COMUNICACIÓN ENTRE LOS USUARIOS Y EL SERVIDOR VIRTUAL

La necesidad de atender y responder las solicitudes de servicio de manera transparente para el usuario requiere de métodos de reenvío de conexiones que hacen uso de las propiedades técnicas y geográficas de la topología de red que se use. Los métodos propuestos por LVS[1] se enumeran a continuación.

6.7.1 REENVÍO DE CONEXIONES POR NAT (VS-NAT)

Este Método aprovecha la posibilidad del kernel de Linux de funcionar como un router con NAT[11] (Network Address Translation), el cual tiene la posibilidad de modificar las direcciones de origen/destino de los paquetes TCP/IP que lo atraviesen: la única dirección real del cluster será la del sistema balanceador; cuando le llegue un paquete modificará la dirección de destino para dirigirla a uno de los servidores reales y la de origen para que le sea devuelto a él, y lo reenviará a la red privada; cuando el servidor real lo procese, se lo envía al sistema balanceador (que es el único punto de salida para todos los equipos del cluster hacia Internet) y éste "deshace" el cambio de direcciones: pone como dirección de origen del paquete con la respuesta suya, y como dirección de destino la del cliente que originó la petición. En la ilustración 2 se puede observar la dinámica del método, extraído de la documentación de Linux Virtual Server, a continuación se listan las actividades del proceso:

- El cliente realiza una petición de servicio, a la IP pública del cluster (la del sistema balanceador de carga).
- El sistema balanceador planifica a qué servidor real va a enviar la petición, reescribe las cabeceras de las tramas TCP/IP y se las envía al servidor.
- El servidor recibe la petición, la procesa, genera la respuesta y se la envía al sistema balanceador de carga.
- El sistema balanceador reescribe de nuevo las cabeceras de las tramas TCP/IP con la respuesta del servidor, y se las envía de vuelta al cliente.
- La respuesta llega al cliente, como si la hubiera generado la IP pública del cluster.

[11] http://es.wikipedia.org/wiki/NAT

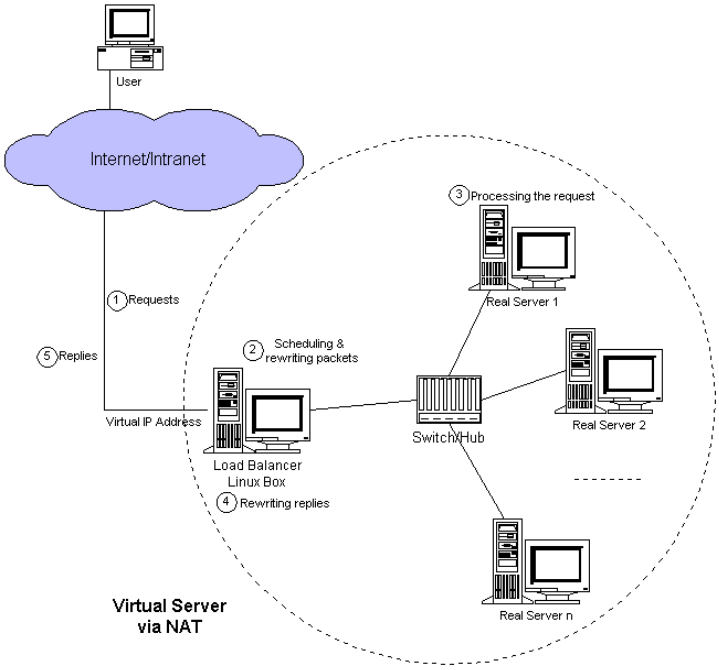


Ilustración 2: Servidor Virtual. Método de reenvío por NAT

Fuente: http://www.linuxvirtualserver.org/VS-NAT.html

6.7.2 REENVÍO DE CONEXIONES POR ENCAPSULADO IP (VS-TUN)

El encapsulado IP consiste en hacer viajar una trama TCP/IP, con sus direcciones de origen y destino, dentro de otra trama con direcciones distintas para, una vez que la trama más externa llegue a su camino, “desencapsular” la trama original y reenrutarla desde allí, como se muestra en la ilustración 3. Es una forma de utilizar un enrutamiento alternativo de una red A a otra red B, forzando un “rodeo” por la red C.

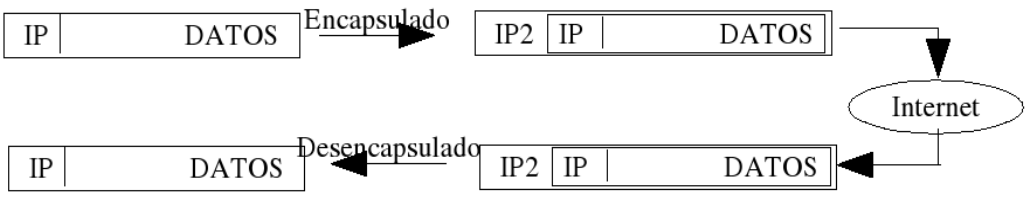


Ilustración 3: IP Tunneling

Fuente: http://www.bisente.com/documentos/clustering/memoria.html

Para utilizar este método de reenvió de conexiones, todos los servidores necesitan tener configurada en alguna interfaz (aún que sea virtual) la IP pública del servidor, y además necesitarán IPs públicas válidas en Internet ya que el punto de entrada al cluster es el sistema balanceador de carga, pero una vez que el tráfico llega a los servidores reales éstos enrutan directamente las respuestas hacia los clientes sin necesidad de pasar de nuevo por el sistema balanceador.

Por otra parte, al realizar la comunicación entre el sistema balanceador y los servidores por medio de encapsulado IP, es posible distribuir los servidores reales a lo largo de una red de área amplia WAN en lugar de tenerlos todos en un mismo segmento de red local.

En la ilustración 4 se puede apreciar el esquema de LVS con encapsulado IP, extraído de la página oficial del proyecto Linux Virtual Server:

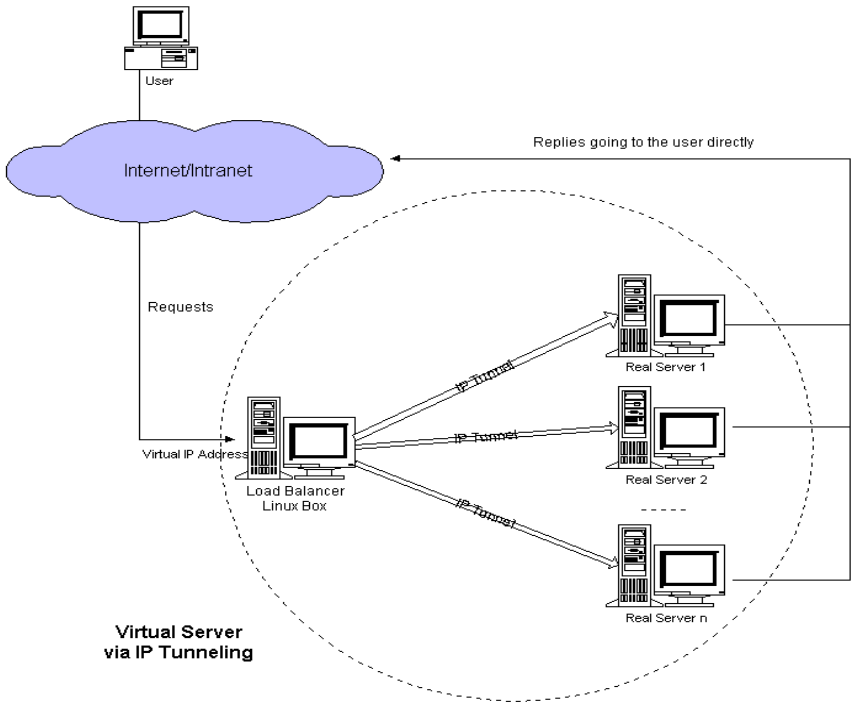


Ilustración 4: Servidor Virtual. Método de reenvío por IP Tunneling

Fuente: http://www.linuxvirtualserver.org/VS-IPTunneling.html

Con esta técnica se evita que el sistema balanceador sea un cuello de botella haciendo que sólo los paquetes de entrada al cluster pasen a través de él, mientras que los de salida los enviará cada servidor real directamente a su destino.

6.7.3 REENVÍO DE CONEXIONES POR ENRUTAMIENTO DIRECTO (VS-DR)

Este método requiere que todos los servidores reales tengan una IP pública, que se encuentren en el mismo segmento físico de red que el sistema balanceador, y además que todos los servidores del cluster (incluido el sistema balanceador) compartan la IP pública del cluster. El sistema balanceador no es un cuello de botella, ya que al igual que en el caso anterior, únicamente pasará a través de él el tráfico en dirección de los clientes al cluster, mientras que el tráfico de salida lo dirigirán directamente los servidores a cada cliente, como se muestra en la ilustración 5.

Todos los nodos tendrán configurado una interfaz con la IP pública del cluster: el sistema balanceador la tendrá en su acceso a Internet y será el punto de entrada al cluster; el resto de nodos estarán conectados al sistema balanceador en la misma red física y en el interfaz conectada a esta red tendrán configurada la IP pública del cluster, configurada de manera que no responda a comandos ARP para no interferir con otros protocolos (todos los equipos responderían por la misma IP con distintas MACs). Cuando llega una petición al sistema balanceador decide a qué servidor enviársela, y redirige el paquete a nivel de enlace (por ejemplo Ethernet) a la dirección MAC del servidor elegido. Cuando llega al servidor con la MAC de destino, este acepta sin más el paquete y genera la respuesta, que enviará directamente al cliente por medio de su IP pública.

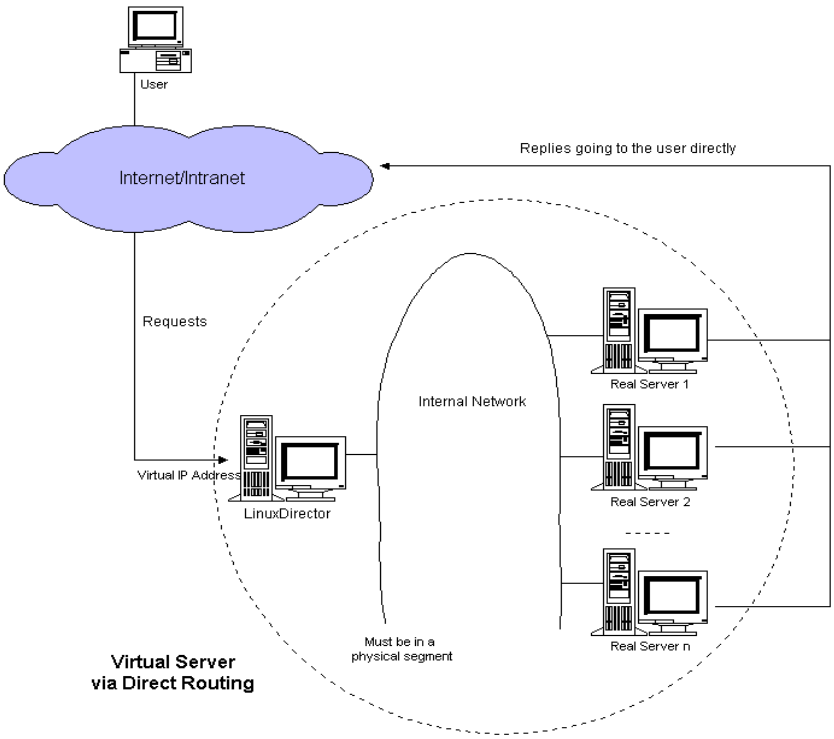


Ilustración 5: Servidor Virtual. Balanceo por enrutamiento directo.

Fuente: http://www.linuxvirtualserver.org/VS-DRouting.html

6.8 SISTEMA DE ARCHIVOS

Un sistema de archivos es un conjunto de tipos abstractos de datos que son necesarios para el almacenamiento, organización jerárquica, manipulación, navegación, acceso y recuperación de los datos que éste almacena.

Los sistemas de archivos modernos diferencian dos tipos de estructuras distintas: archivos y directorios. Una de las características principales de un sistema de archivos es el comportamiento o semántica, cuando más de un proceso accede a la misma sección de un archivo.

Los sistemas de archivos resaltan en sus diseños el rendimiento y la escalabilidad para funcionar tanto en una docena de equipos como para varios millares manteniendo niveles altos de ancho de banda en la Entrada/Salida. Por lo general

los sistemas centralizados no se escalan bien, en cierto momento cuando el sistema crece demasiado se puede convertir en cuello de botella. Otro aspecto de relevancia en los diseños de los sistemas de archivos es la integridad de la información que está disponible en una organización, la cual debe ser completa en su conjunto y no debe ser alterada por personas no autorizadas. Además la disponibilidad de la información es tan importante como el rendimiento, la escalabilidad y la integridad, esto significa que la información solicitada o requerida por los usuarios autorizados siempre debe estar disponible para ser usada.[10]

6.8.1 SISTEMAS DE ARCHIVOS DISTRIBUIDOS

Con la aparición de las redes de interconexión, surgió la necesidad de compartir datos entre diferentes máquinas debido a la economía o la naturaleza de algunas aplicaciones, situación que originó la aparición de los sistemas de archivos distribuidos (DFS), estos son una implementación distribuida del clásico modelo de tiempo compartido de un sistema de archivos, donde varios usuarios comparten archivos y almacenan recursos [9]. NFS [11], Coda [12] o GFS [13] son ejemplos de sistemas de archivos distribuidos.

6.8.1.1 NFS

NFS (Network File System) fue diseñado originalmente por Sun Microsystems en 1985 [11]. NFS permite el acceso transparente a archivos y directorios situados en máquinas remotas, para poder utilizarlos como si fueran locales. Sigue un esquema cliente-servidor, de forma que el nodo remoto se denomina servidor NFS y el nodo local se denomina cliente NFS. El servidor sitúa directorios de su propio árbol de directorios a disposición de otros nodos que se encuentran conectados a través de una red. Para llevar a cabo esta tarea, el servidor debe exportar dichos directorios. Los clientes con la finalidad de utilizar los directorios exportados previamente deben “montarlos” en algún directorio de su propio sistema de

archivos, directorio al cual se denomina “punto de montaje”.

6.8.1.2 CODA

CODA (Constant Data Availability) es un sistema de archivos implementado en la Universidad Carnegie Mellon y descendiente del sistema de archivos AFS. Con el objetivo de mejorar la disponibilidad en Coda pueden existir múltiples copias de cada archivo en diferentes servidores. De este modo, proporcionar tolerancia a fallos al uso de los archivos, eliminando el impacto que supone que un servidor deje de funcionar.[12]

6.8.2 SISTEMA DE ARCHIVOS PARALELOS

En los sistemas de archivos distribuidos, cada archivo se almacena en un servidor, y el ancho de banda de acceso a un archivo se encuentra limitado por el acceso a un único servidor, lo que convierte a los servidores en un cuello de botella en el sistema.

Este problema[14], es originado por el desequilibrio existente entre el tiempo de computo y el tiempo de E/S(Entrada/Salida). Mientras que el rendimiento de los procesadores y de los sistemas de computación se ha incrementado de forma drástica en los últimos años, la velocidad de acceso de los sistemas de almacenamiento no ha seguido la misma evolución.

Esta diferencia es aún más latente en sistemas multiprocesadores, donde, se multiplica el rendimiento del procesador por el número de procesadores del sistema.

Las propuestas de solución más relevantes para resolver este problema son:

- Utilizar paralelismo en el sistema de E/S con la distribución de los datos de un archivo entre diferentes dispositivos y/o servidores.
- Emplear sistemas de almacenamiento de altas prestaciones (redes de

almacenamiento)[12].

El paralelismo en la E/S de sistemas de archivos procede de la aparición del sistema RAID (Redundant Array of Inexpensive Disks), descritos inicialmente en los años 80 [15][16]. Consiste en crear un array (arreglo) de varios discos simples ("inexpensive", baratos), y tratarlos como un todo a la hora de acceder a ellos, dependiendo del nivel de RAID que se implemente, ofrecen mejor rendimiento y mayor grado de tolerancia a fallos que un único disco, debido a que distribuyen los datos a través del conjunto de discos para almacenar la información y si fuera necesario datos redundantes para recuperarse de fallos físicos de los dispositivos.

La distribución de datos mediante el empleo de particiones distribuidas, permite que un único archivo se distribuya entre varios discos servidores de E/S. Esta solución permite acceder en paralelo a los datos de un mismo archivo, algo que no se puede conseguir con el empleo de sistemas de archivos distribuidos tradicionales. Algunas de los sistemas de archivos paralelos son PVFS [17], Lustre File System[19] y GPFS[20] entre otros.

6.8.2.1 PARALLEL VIRTUAL FILE SYSTEM PVFS

PVFS2 provee de un sistema de archivos en red paralelo de alta eficiencia y escalable, normalmente utilizado en entornos de cluster. PVFS2 es un proyecto de Software Libre que no requiere hardware especial o modificaciones en el núcleo para que funcione. Está compuesto por un espacio para los metadatos (MGR) que puede ser uno o varios nodos, el conjunto de servidores de almacenamiento (Input/Output nodes) y los clientes del sistema de archivos (Compute nodes) como se muestra en la ilustración 6. Algunas características de este sistema de archivos son:

- Provee a los usuarios un espacio de nombres consistente entre los nodos del cluster que permite a los programadores acceder a los archivos desde múltiples nodos.

[12] http://es.wikipedia.org/wiki/Almacenamiento_asociado_a_red

- Distribución física de los datos entre los discos de los nodos que permite evitar cuellos de botella tanto en la interfaz del disco como así también en la red proporcionando mayor ancho de banda a los recursos de Entrada/Salida (E/S).
- Acceso transparente para las utilidades existentes (ls, cd, etc.)
- Alto rendimiento en espacio de almacenamiento para las aplicaciones.

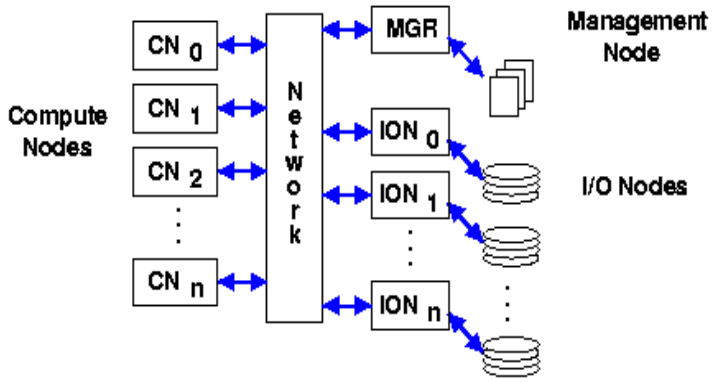


Ilustración 6: Arquitectura de PVFS2

Fuente: http://www.parl.clemson.edu/pvfs/index.html

PVFS2 provee un mismo espacio de nombre para todo el cluster y es accesible por las utilidades habituales. PVFS2 se monta en todos los nodos y en el mismo directorio simultáneamente, permitiendo el acceso simultáneo a todos los archivos del sistema PVFS2, a través del mismo esquema de directorios. Una vez que el sistema está montado, se puede trabajar con las herramientas típicas, como ls, cp y rm.

Para conseguir un alto rendimiento en el acceso a los datos concurrentemente, PVFS2 distribuye los datos en múltiples nodos del cluster, denominados I/O nodos. Por medio de libpvfs los clientes de PVFS2 acceden al MGR para solicitar la información referente a un archivo, seguidamente la comunicación se hace directamente con los I/O nodos de forma paralela, en los cuales se encuentra almacenada la información, como se muestra en la ilustración 7. Distribuyendo los datos en múltiples nodos, los clientes poseen diferentes rutas hacia los datos,

eliminado de esta forma los cuellos de botella (bottlenecks) y mejorando el ancho de banda para múltiples clientes [17].

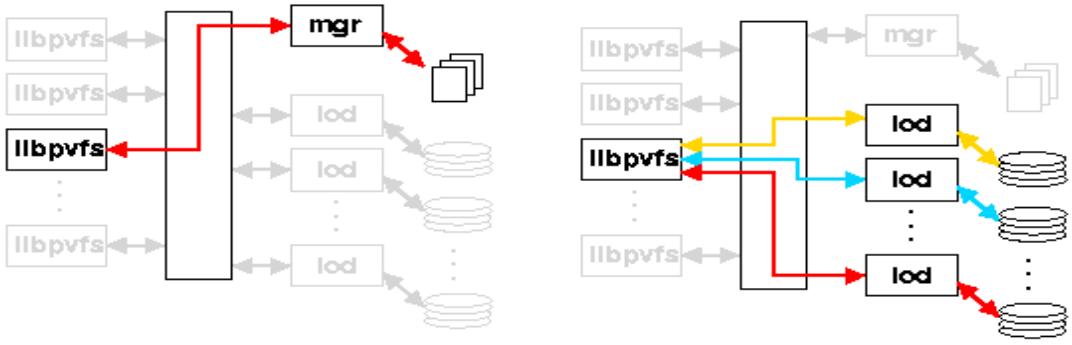


Ilustración 7: Dinámica de funcionamiento de PVFS2

Fuente: http://www.parl.clemson.edu/pvfs/index.html

6.8.2.2 LUSTRE FILE SYSTEM

Es un sistema de archivos para cluster iniciado por Carnegie Mellon University en el año 1999, proponiendo una arquitectura basada en objetos. Cuenta con cuatro componentes: el cliente, que es usado para acceder al sistema de archivos, el sistema de gestión (MGS) en el cual se define la información acerca del sistema de archivos, el servidor de almacenamiento de Objetos (OSS), que provee el servicio I/O, y servidores de Metadatos (MDS), los cuales manejan los nombres y directorios dentro del sistema de archivos [13]. El total de almacenamiento manejado por un OSS es separado en volúmenes, la capacidad de cada volumen varia de 2 a 8 Terabytes. Cada OSS está encargado de manejar múltiples Object Storage Target (OST), uno por cada volumen.

Al abrir un archivo, el cliente contacta al MDS para obtener la información del archivo, y las operaciones subsiguientes se realizarán con el OSS que contenga el archivo, como se muestra en la ilustración 8. No existe referencia directa a los datos. [19]

Los servidores de metadatos guardan un historial de sus transacciones,

guardando cambios en la metadatos y en los estados del cluster. Esto reduce el tiempo de restauración del sistema de archivos.
Otras características son:

- Compatible con Posix.
- Soporta varias distribuciones de GNU/Linux.
- Caching en los MDS.
- Configuración e información de estado en formato XML y LDAP.
- Cada OST maneja los locks de los archivos que contiene. Locking de archivos distribuido.
- Soporta múltiples tipos de redes.
- Cuenta con herramientas de respaldo y registro de estado snapshots del sistema.

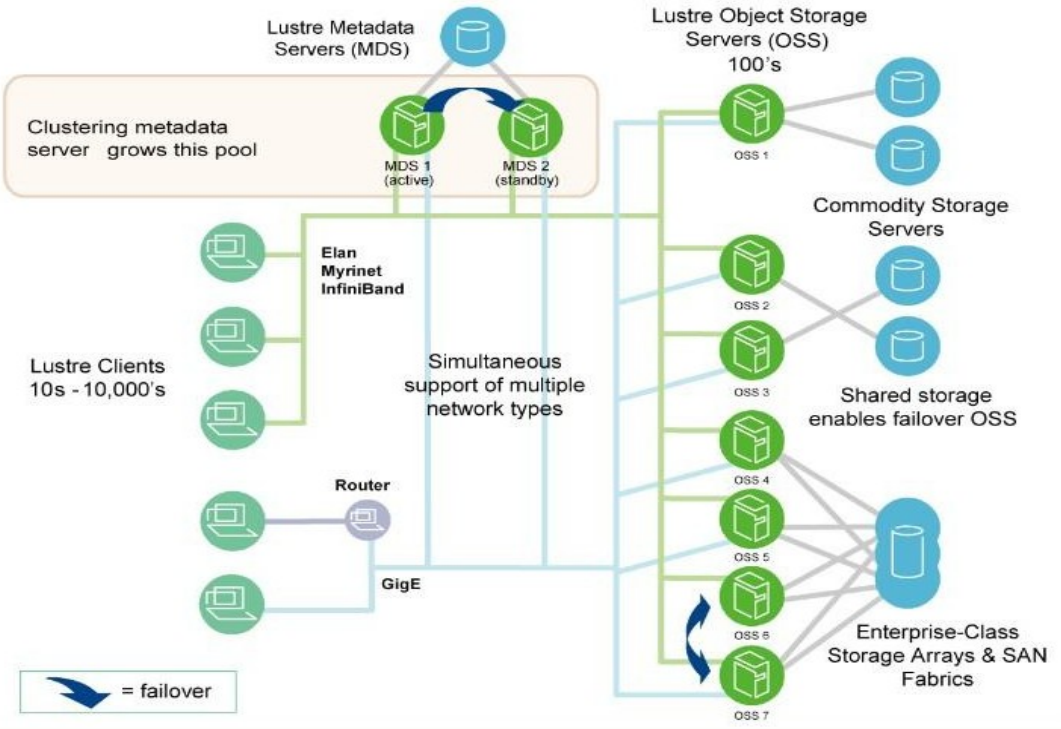


Ilustración 8: Arquitectura de Lustre File System

Fuente: Lustre 1.6 Operations Manual Capitulo 1

6.8.3 TOLERANCIA A FALLOS EN SISTEMAS DE ARCHIVOS

Tolerancia a fallos es la capacidad de un sistema de mantenerse en funcionamiento ante un fallo. Los problemas son circunstancias no deseadas (aunque no inesperadas) que reducen la garantía de funcionamiento, y hacen que no se pueda confiar en el servicio suministrado.

Una *avería* se presenta cuando el servicio prestado por el sistema no es el especificado o hay cambios que hacen que el usuario aprecie que el sistema no funciona bien. Un *error* es la señal de un estado interno incorrecto del sistema. Un *fallo* es un defecto o imperfección física en el hardware o software del sistema[54], la 9 resume la relación entre fallos, errores y averías.

Ilustración 9: Simplificación de la relación entre fallos, errores y averías

Fuente: Tolerancia a fallos, en clusters de computadores geográficamente distribuidos, basada en Replicación de Datos, Josemar Rodrigues de Souza

La redundancia como alternativa de solución está íntimamente relacionada con la tolerancia a fallos, ya que mediante esta técnica se evita los SPOF[13] y puede existir en varios niveles de redundancia: hardware, información, software.

6.8.3.1 DRBD

Un modulo del kernel de GNU/Linux que está relacionado con la redundancia a nivel de información es DRBD (Distributed Replicated Block Device), el cual se encarga de la denominación de bloques lógicos que son administrados por el software, permite hacer replicación remota en tiempo real en dispositivos de almacenamiento por medio de TCP/IP (RAID 1 en red), en la 10 se muestra un ejemplo de una configuración Primary/Secondary en la cual los 2 discos duros son identificados por el sistema operativo con un solo dispositivo físico (/dev/drbd0), además DRBD es fácilmente integrable con software para la administración y monitorización de servicios (como Heartbeat del Proyecto Linux-HA) y con sistemas de archivos paralelos como Lustre, PVFS2 o distribuidos como NFS de tal forma que se tenga rendimiento en la E/S, tolerancia por replicación de los datos y monitorización del correcto funcionamiento de los servicios de almacenamiento.

[13] SPOF http://es.wikipedia.org/wiki/SPOF

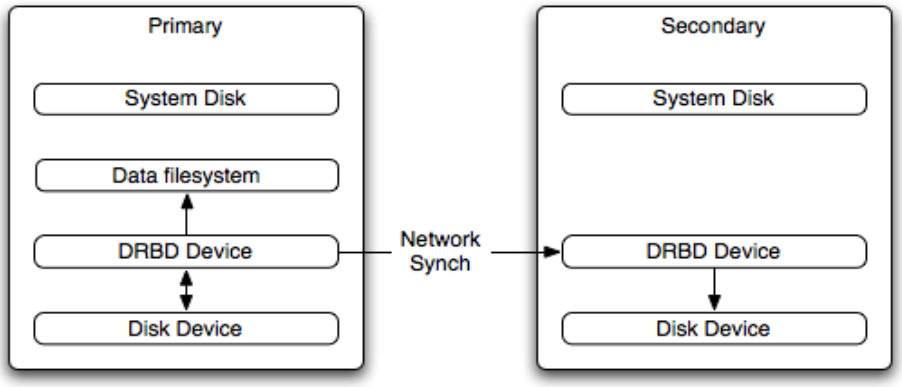


Ilustración 10: Ejemplo de configuración Primary/Secondary

Fuente: Chapter 14. High Availability, Scalability, and DRBD
http://www.opensourcedocs.com/mysql/ha-overview.html

6.8.4 SISTEMAS DE ARCHIVOS PARALELOS CON REPLICACIÓN EN TIEMPO REAL

El empleo de paralelismo en el sistema de archivos es diferente al empleo de sistemas de archivos replicados. En un sistema de archivos replicado, cada disco (en cada servidor) almacena una copia completa de un archivo. Utilizando E/S paralela, cada disco (en cada servidor) almacena una parte del archivo, lo que permite acceder al archivo en paralelo. La ilustración 11 muestra la diferencia entre estos dos enfoques.

Para concluir, el empleo de paralelismo en el sistema de E/S puede realizarse sobre un sistema distribuido compuesto por varias máquinas con discos conectados, lo que permite incrementar el ancho de banda del sistema de E/S. El paralelismo se consigue con la distribución de los datos entre varios servidores y discos, lo que permite leer (y escribir) los datos de los archivos en paralelo desde diferentes servidores.

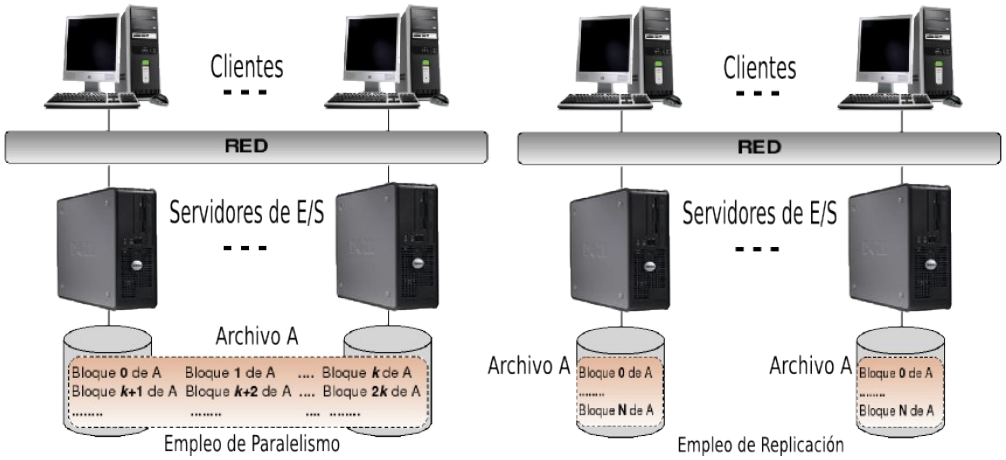


Ilustración 11: Diferencias entre replicación y paralelismo.

Fuente: Arquitectura multiagente para E/S de alto rendimiento en clusters. Pérez Hernández, María de los Santos.

La unión del paralelismo y la replicación da como resultado un sistema de archivos de alto rendimiento en la E/S y tolerante a fallos de los discos o software del sistema de archivos. Tener replicación en tiempo real demanda uso de la red y dependiendo del protocolo que se use en DRBD se tendrá un tiempo de penalización en la escritura de archivos por el uso de la replicación, la ilustración 12 muestra una modificación de la ilustración 11, uniendo estas dos técnicas.

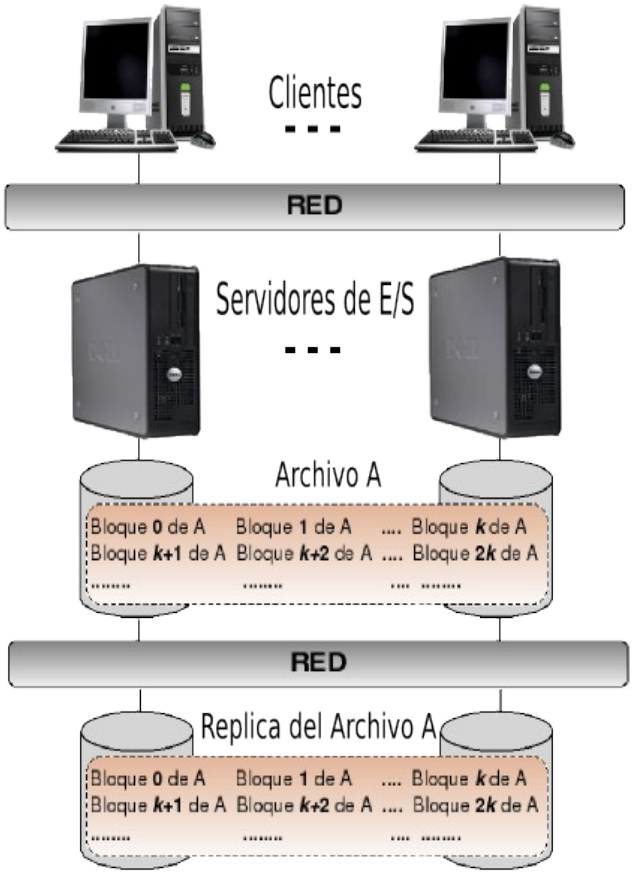


Ilustración 12: Unión de replicación y paralelismo

Fuente: Autores

6.9 DISPONIBILIDAD EN SISTEMAS DISTRIBUIDOS

Un dilema común en las TIC's es cómo sobrevivir a interrupciones planeadas y no planeadas, por medio de los los sistemas distribuidos se logra distribuir la probabilidad de fallos, en la práctica, esto se requiere del cuidado en el diseño, configuración e integración de los componentes que hacen parte del mismo.

En un entorno distribuido, los criterios básicos para calcular la disponibilidad son los siguientes[24]:

- Determine la disponibilidad de cada componente, como un número de 0 a 1.

- Multiplicar la disponibilidad de todos los componentes juntos para conseguir la disponibilidad total, que generalmente se expresa como un porcentaje.

En un cluster de alta disponibilidad se pueden identificar 3 componentes que conforman una arquitectura de este tipo, los cuales son: balanceadores de carga, nodos del servicio prestado y el almacenamiento compartido.

Matemáticamente, esto se describe como[24]:

$$D_{Total} = D_{Balanceodecarga} * D_{Nodosdeservicio} * D_{Almacenamientocompartido} \quad (2)$$

$$D_{Total\%} = D_{Total} * 100\ \% \quad (3)$$

$$D = Disponibilidad$$

En el contexto de disponibilidad existen dos formas de clasificar los componentes computacionales de un cluster, los cuales son:

- Cluster de nodos Activos: Conjunto de componentes computacionales que se encuentran prestando su servicio activamente todo el tiempo.
- Cluster de nodos Activo/Pasivo: Conjunto de componentes computacionales que tienen un componente igual asociado, se agrupan en parejas Activo/Pasivo, en la cual un nodo presta sus servicios activamente y el otro nodo lo monitoriza, en caso de fallas el nodo pasivo cambiara como activo y retoma los servicios del primero.

6.9.1 FALLO DEL CLUSTER

Considerando un cluster de nodos Activo/Pasivo, si falla un nodo el otro retomara los servicios, sin embargo si fallan los dos nodos el servicio estará no disponible. Se definen los siguientes parámetros para el cálculo de la disponibilidad del sistema.

n = número de nodos del sistema

a = disponibilidad de un nodo (porcentaje del tiempo que está funcionado correctamente)

f = probabilidad de fallo de un nodo (porcentaje del tiempo que no funciona)

F_d = probabilidad de fallo del sistema

F = probabilidad de que el sistema este dañado

A = disponibilidad del sistema

Si a es la probabilidad de que un nodo esté funcionando correctamente y f es la probabilidad de que un nodo no esté prestando el servicio, entonces se puede decir que $f = (1 - a)$.

La probabilidad de que 2 nodos no estuvieran prestando el servicio al mismo tiempo es la probabilidad de que un nodo no funcione y la probabilidad que el segundo nodo tampoco funcione, matemáticamente esto es $f = (1-a)^2$, enmarcado en un cluster de n nodos seria $f = (1-a)^n$. El número de formas (c) en las que pueden fallar los componentes del cluster depende de la manera como fue organizado, mas adelante se analiza la variable c para cada caso especifico en la organización de los componentes del cluster de alta disponibilidad.

Por ultimo la probabilidad de fallo del cluster es[23]:

$$F_d = c * (1 - a)^n \quad (4)$$

6.9.2 EVENTOS DE FAILOVER

En el caso de que un nodo falle el sistema se mantiene en funcionamiento, sin embargo hay un tiempo usado para la retoma de servicios, este es el tiempo que transcurre desde que un nodo activo es reportado como dañado, hasta que su copia (nodo pasivo) se convierte en activo y retoma los servicios.

Se define lo siguiente:

F_f = Probabilidad que el sistema no funcione por causa de un proceso de failover.

$MTFO$ = Tiempo medio de failover (tiempo promedio de retoma de servicios en caso de fallo)

$MTBF$ = Tiempo medio entre fallos de un nodo.

La proporción de tiempo que el sistema esté dañado durante un failover de un nodo en particular es $MTFO / MTBF$, teniendo en cuanta que hay n nodos en el sistema, el fallo de cualquier nodo llevara asociado un evento de failover. La probabilidad que el sistema no funcione durante el proceso de failover es[23]:

$$F_f = n \frac{MTFO}{MTBF} \quad (5)$$

6.9.3 DISPONIBILIDAD EN UN CLUSTER

La probabilidad que el sistema no funcione (F) es la probabilidad de fallo del cluster (F_d) mas la probabilidad que el sistema no funcione por causa de un proceso de failover (F_f) [23]:

$$F = F_d + F_f = c * (1 - a)^n + n \frac{MTFO}{MTBF} \quad (6)$$

La disponibilidad del sistema (D) es:

$$D = 1 - F \quad (7)$$

7. DISEÑO DE LA ARQUITECTURA PLANTEADA

Para el diseño del servicio de correo basado en un cluster de alta disponibilidad, según la revisión bibliográfica se encontró que la arquitectura provista por el proyecto LVS (Linux Virtual Server) integra todos los componentes necesarios organizados en tres capas de funcionamiento las cuales son: monitoreo de servicios y balanceo de carga, servicio distribuido y almacenamiento compartido como se muestra en la ilustración 13. La arquitectura propuesta en este trabajo está basada en la integración de las capas anteriormente mencionadas las cuales serán descritas en las secciones siguientes.

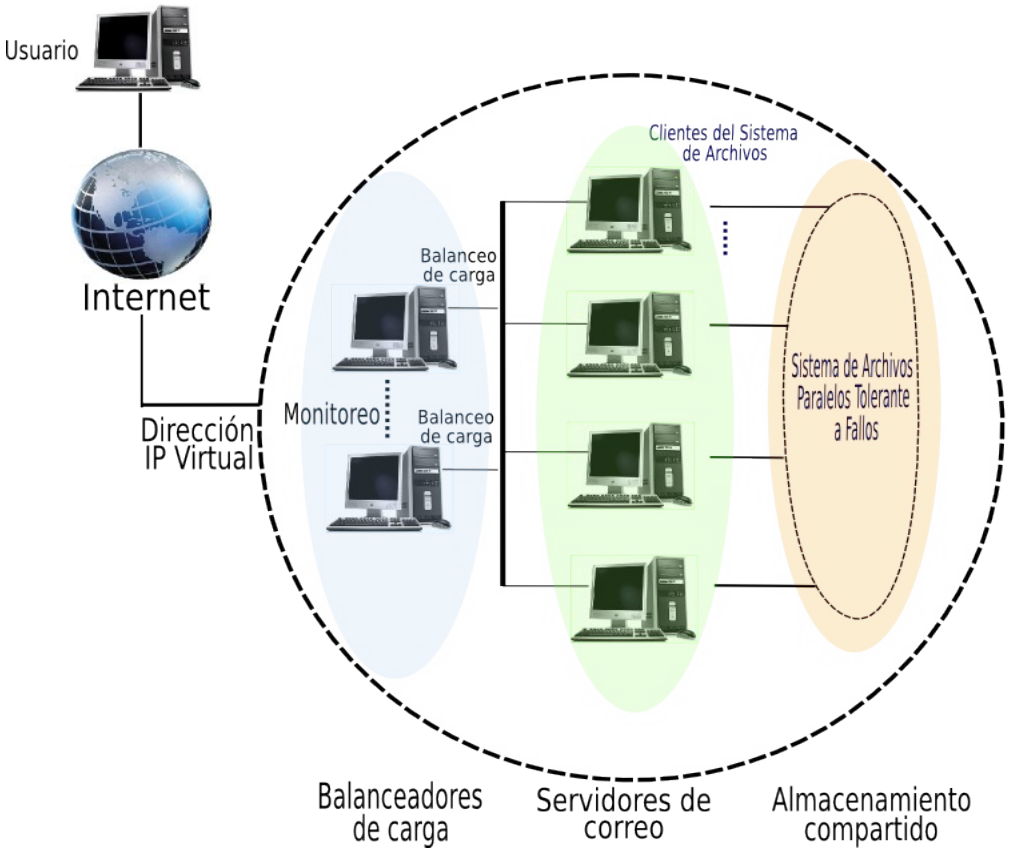


Ilustración 13: Diseño planteado

Fuente: Autores

7.1 CAPA DE MONITOREO DE SERVICIOS Y BALANCEO DE CARGA

Esta es la primera de las capas, por lo tanto, es la encargada de recibir las peticiones de servicio y de repartirlas en la granja de servidores reales usando algún algoritmo de planificación por medio de Ldirectord. Su probabilidad de fallo se distribuye al estar compuesta por dos computadores quienes están en continua monitorización de los servicios prestados por el servidor virtual. La monitorización de los servicios prestados por el cluster está a cargo de heartbeat configurado de forma Activo/Pasivo, lo que significa que uno de los 2 estará como principal, procesando las peticiones de servicio de los usuarios y de otros servidores de correo que quieran establecer comunicación para el envío de mensajes. El otro nodo denominado secundario, estará esperando que el principal falle para retomar los servicios que se estaban prestando antes del suceso siendo este proceso transparente para el usuario.

7.2 CAPA DE SERVICIO DE CORREO

En la capa de servicio se encuentran los servidores reales que son cuatro nodos configurados idénticamente como servidores de correo para el manejo de los protocolos SMTP(con sus extensiones) para el envío y recepción de mensajes de correo y el protocolo IMAP para el acceso a los buzones de correo de los usuarios.

7.2.1 DESCRIPCIÓN DE LAS HERRAMIENTAS SOFTWARE USADAS PARA EL SERVICIO DE CORREO

En esta sección se hace una descripción de las herramientas usadas para la implementación del diseño del servicio de correo electrónico en el cluster de alta disponibilidad, empezando por el MTA Postfix, el servidor Courier para el manejo del protocolo IMAP, los métodos de autenticación para los usuarios, la interfaz usada para el acceso al servicio de correo Roundcube Webmail, y los plugins

para el manejo tanto del spam como de los virus en el servidor.

7.2.1.1 POSTFIX

El MTA usado para la implementación del diseño para el servicio de correo electrónico en el cluster de alta disponibilidad es *postfix*[14]. Postfix tiene como principal característica su diseño modular y se ha constituido en una alternativa robusta al lado de otros MTA's como sendmail y qmail por su fácil integración con otras herramientas como bases de datos MYSQL, directorios mediante LDAP, con tecnologías como SASL, LMTP etc. Su funcionamiento se basa en algunos procesos que se comunican a través de sockets que se crean, por razones de seguridad, en un directorio de acceso restringido. La información que intercambian los diversos procesos es la mínima posible, limitándose en la mayoría de los casos a la referencia de la entrada en una cola y la relación de destinatarios, o a un simple identificador de estado.

7.2.1.2 SERVIDOR COURIER-IMAP

El servidor Courier IMAP fue el elegido para realizar el acceso de los usuarios a los buzones de correo empleando el protocolo IMAP. Las características más importantes de este servidor son: su sencillez, su velocidad y la escalabilidad al estar en capacidades para manejar cientos de miles de cuentas. Además Permite integración con múltiples MTA's como postfix, sendmail, qmail, entre otros, y usa Maildir como formato para los buzones de correo.

La autenticación para los usuarios de correo electrónico con el servidor Courier-IMAP es por medio del mecanismo PAM (“Pluggable Authentication Modules'', o Módulos de Autenticación Enlazables”) el cual proporciona una interfaz entre nuestro servidor de correo con lo diferentes métodos de autenticación. Cada programa que utilice PAM definirá el nombre de su propio servicio', por lo que para nuestro caso el servicio se llamará *imap,* ya que es el nombre del protocolo que se

[14] http://www.postfix.org/

va a usar para acceder a los buzones de correo de cada usuario. El método de autenticación en el servidor es el clásico en los sistemas UNIX donde cada usuario posee un nombre de entrada al sistema o login y una clave o password; ambos datos se almacenan generalmente en el fichero /etc/passwd. Para cifrar las claves de acceso de sus usuarios, el sistema operativo Unix emplea un sistema de criptografía irreversible que utiliza la función estándar de C crypt.

7.2.1.3 FORMATO MAILDIR

En el servidor de correo implementado se usa este formato para los buzones de correo. Este formato de spool de correo electrónico no bloquea los ficheros para mantener la integridad del mensaje, porque los mensajes se almacenan en ficheros distintos con nombres únicos. Maildir es un directorio (usualmente llamado Maildir) con tres subdirectorios llamados tmp, new, y cur, donde todos los subdirectorios deben residir en el mismo sistema de archivos.

7.2.1.4 SPAMASSASSIN

SpamAssassin es un filtro de correo que trata de identificar el spam mediante el análisis del texto y el uso en tiempo real de algunas listas negras a través de Internet. A partir de su base de datos de reglas, utiliza un amplio abanico de pruebas heurísticas en las cabeceras y el cuerpo de los correos para identificar el spam, también conocido como *“correo electrónico comercial no solicitado”* logrando identificar acertadamente entre un 95 y un 99% del spam en Internet . Una vez identificado, el correo puede ser opcionalmente marcado como spam o más tarde filtrado usando el cliente de correo del usuario.

7.2.1.5 CLAM ANTIVIRUS

ClamAV es una herramienta antivirus GPL para UNIX. El propósito principal de este software es la integración con los servidores de correo en la labor de escaneo de datos adjuntos en los mensajes que los transitan. Como características destacables está el soporte de firmas digitales en la actualización de la base de datos (proceso automático), el análisis durante el acceso bajo GNU/Linux, la detección de más de 20.000 virus, gusanos y troyanos, el soporte integrado para archivos comprimidos con Rar, Zip, Gzip y Bzip2 y formatos de correo Mbox, Maildir. En el servidor de correo no se permite el envío de archivos ejecutables o todo aquel archivo que se considere malicioso.

7.2.1.6 AMAVIS-NEW

Amavisd-new es una interfaz de alto rendimiento y fiabilidad entre el MTA y uno o más filtros de contenidos: antivirus y el software antispam. Está escrito en Perl, asegurando alta fiabilidad, portabilidad y facilidad de mantenimiento. Se comunica con el MTA vía ESMTP y no existen problemas de sincronización en su diseño que pudieran causar pérdidas de correos.

7.3 CAPA DE ALMACENAMIENTO COMPARTIDO

7.3.1 SELECCIÓN DEL SISTEMA DE ARCHIVOS PARALELO CON REPLICACIÓN EN TIEMPO REAL.

Con las características antes nombradas en el marco teórico, se pudo elegir los sistemas de archivos que cumplieron con el paralelismo en la E/S y que además permitieron la integración con DRBD para tener replicación de la información. Es así como se decide comparar el rendimiento en la E/S en varios escenarios en los sistemas de archivos PVFS2 con DRBD y Lustre File System con DRBD, la realización de estas pruebas se llevará a cabo con benchmarks que se ejecutan

desde un cluster MPICH2[15] el cual es cliente del sistema de archivos y hace operaciones de E/S sobre el mismo. Como característica para la elección del sistema de archivos paralelo se tuvo en cuenta que la licencia fuera de uso libre.

7.3.1.1 PRUEBAS REALIZADAS

Los benchmark usados para la realización de las pruebas están basados en la librería ROMIO[18] de MPI IO, estas funciones permiten que los benchmark perf y mkrandpfile usen funciones de lectura/escritura en paralelo como se describe en el anexo E.

Las pruebas realizadas miden el rendimiento de E/S de los sistemas de archivos, se realizó para tamaños de archivos de 10K, 100K, 500K, 2M, 10M, 100M, 500M, ejecutando corridas en paralelo de 2, 3, 4 y 5 procesadores. La prueba se hizo desde un cluster con MPICH2 el cual accedía al sistema de archivos como cliente, separando los usuarios del sistema de archivos de los servidores de almacenamiento.

Se plantea un escenario que sea muy aproximado a la realidad y pueda reflejar el desempeño de los sistemas de archivos en dos eventos: E/S en archivos concurrente (usando perf) desde n nodos a 1 archivo y E/S archivos separados (usando mkrandpfile) desde n nodos a n archivos como se puede observar en la ilustración 14.

[15] http://www.mcs.anl.gov/research/projects/mpich2/

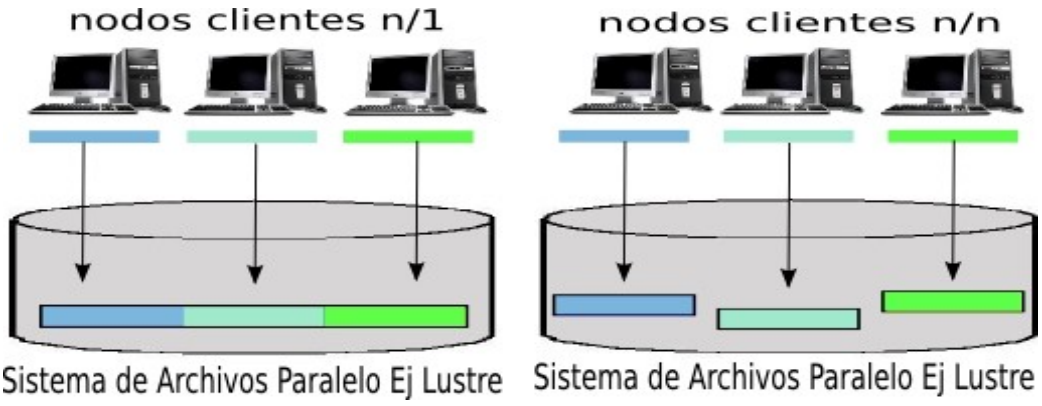


Ilustración 14: E/S concurrente n nodos sobre 1 archivo y E/S separada n nodos sobre n archivos separados

Fuente: Autores

Los sistemas de archivos PVFS2 y Lustre realizan replicación de cada uno de sus nodos servidores de almacenamiento y/o nodo de metadatos con DRBD.

7.3.1.2 RESULTADOS

La primera prueba que se hizo fue la de realizar operaciones de E/S sobre un mismo archivo, el benchamak perf se ejecutó varias veces variando el tamaño de archivo que sería accedido en común por el cluster de clientes del sistema de archivos. La organización de Lustre File System está compuesta por 1 equipo como MDT y MGS, 3 equipos que aportaban 2 OSS de 70GB cada uno, para un total de aproximadamente 420 GB de almacenamiento. Cada uno de los componentes de Lustre está replicado mediante DRBD.

PVFS2 se organizó usando 4 equipos como servidores de almacenamiento y uno de esos como metadato, para un total de aproximadamente 420 GB, cada servidor está replicado mediante DRBD. La infraestructura de red usada fue la del CENTIC, la cual está dada por tarjetas Fast Ethernet en los equipos que se encuentran conectados a un switch Fast Ethernet. A continuación se explica el comportamiento del sistema de archivos cuando se realizó la prueba con 4 nodos como clientes de Lustre.

El benchmark perf ejecutado sobre Lustre refleja un comportamiento alto en el ancho de banda de E/S en tamaños inferiores a 2 MB en el archivo creado, escribiendo 500KB por cada uno de los 4 nodos sobre un solo archivo, con la unión de los cuatro nodos se tiene un archivo concurrente de 2000KB que permite tener un ancho de banda máximo de aproximadamente 4625 MB/s en la lectura, como resultado de implementar Lustre con 6 OST y la utilización de striping[16], permitiendo que partes del archivos se almacenen en los diversos OST. La ilustración 15 muestra los resultados del ancho de banda de E/S que se obtuvieron.

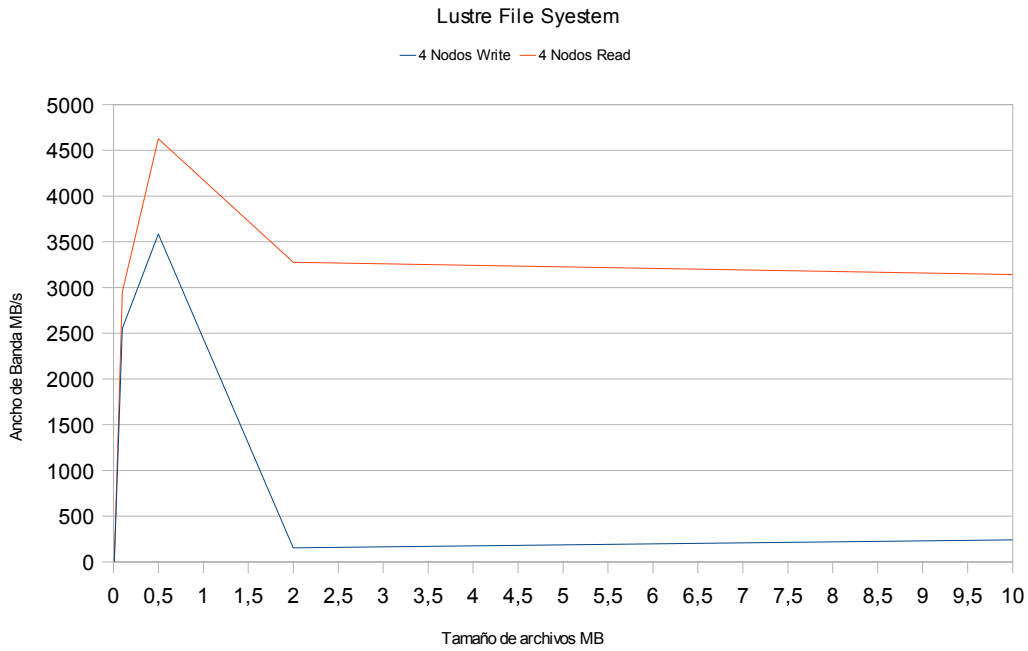


Ilustración 15: Lectura y escritura desde 4 nodos sobre 1 solo archivo usando el sistema de archivos Lustre.

Fuente: Autores

El comportamiento de PVFS2 al ser ejecutado el benchmark perf refleja que el ancho de banda de E/S para archivos pequeños (< 1MB) fluctúa y no es muy clara su tendencia, para archivos mayores a 1,5 MB la tendencia es creciente. Esta prueba se realizó escribiendo 500KB por cada uno de los 4 nodos clientes sobre

[16] Subcapítulo 1.3.1 Lustre File System and Striping
http://manual.lustre.org/manual/LustreManual16_HTML/ClusterWithLustre.html

un solo archivo y los resultados se muestran en la ilustración 16.

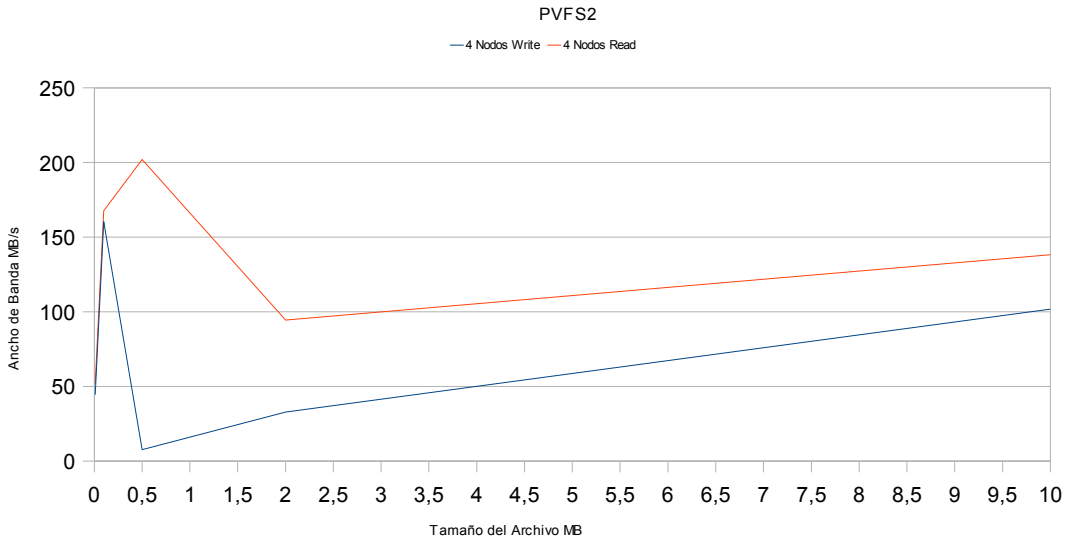


Ilustración 16: Lectura y escritura desde 4 nodos sobre 1 solo archivo usando el sistema de archivos PVFS2.

Fuente: Autores

Al comparar la escritura concurrente entre los dos sistemas de archivos se puede observar que Lustre tiene un mejor comportamiento del ancho de banda para archivos pequeños (< 1,5 MB), a medida que el tamaño del archivo crece los 2 sistemas de archivos tienden a igualar el ancho de banda, también se puede ver que se hace uso mas intensivo de la infraestructura de red y de los recursos de computo es por eso que el rendimiento tiende a equilibrarse. Esta comparación se puede ver en la ilustración 17.

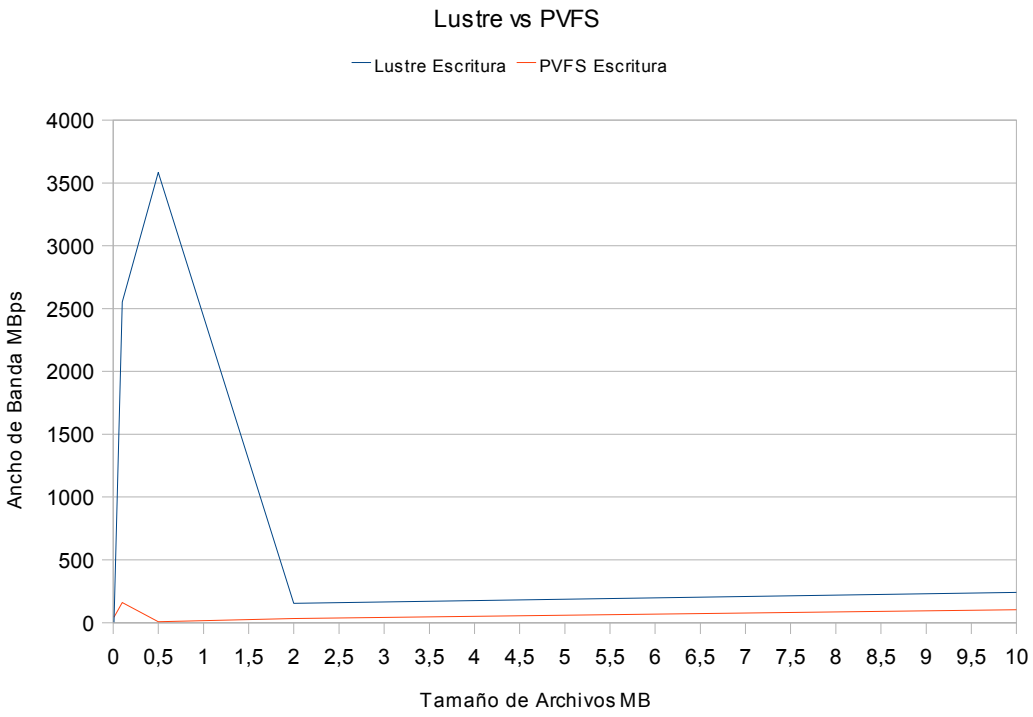
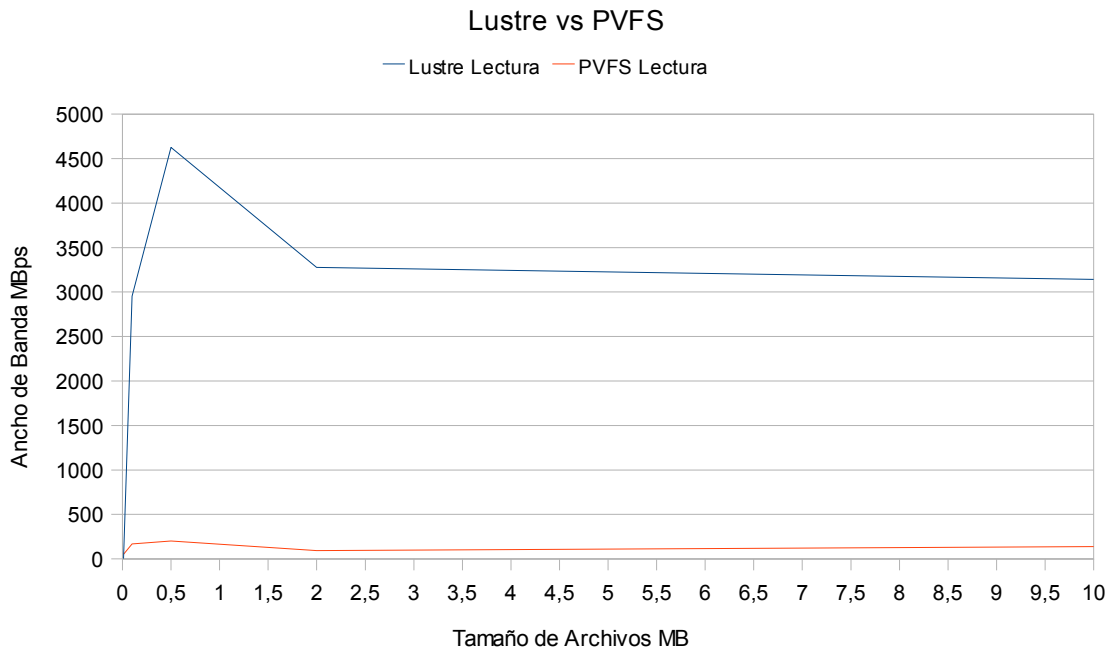


Ilustración 17 Comparación de escritura desde 4 nodos sobre 1 solo archivo entre PVFS2 y Lustre Fuente: Autores

En la Lectura del archivo concurrente es clara la superioridad de Lustre sobre PVFS2, debido a las propiedades de Lustre de tener MDC (Metadata Client) y OSC (Object Storaege Client) en cada uno de los 4 nodos clientes del sistema de archivos, por medio de *Lockless I/O* permite tener metadatos y archivos actualizados del lado del cliente, otro factor que mejora la lectura la utilización de la memoria RAM de los OSS como *caching* de solo lectura para los archivos almacenados. Los resultados de esta comparación se muestran en la ilustración 18.

Ilustración 18: Comparación de lectura desde 4 nodos sobre 1 solo archivo entre PVFS2 y Lustre.



Fuente: Autores

La segunda prueba que se llevo a cabo fue la de realizar operaciones de E/S sobre archivos separados, el benchamrk mkrandpfile se ejecutó varias veces cambiando el tamaño de los archivos creados y leídos por cada nodo del cluster de clientes. La organización de los sistemas de archivos fue la misma que se uso en la anterior prueba.

La ejecución del benchmark mkrandpfile sobre Lustre refleja el alto desempeño en archivos pequeños que se obtiene en la escritura de 4 archivos separados, a medida que se aumenta el tamaño de los archivos el ancho de banda tiende suavemente a disminuir. Los resultados de este comportamiento se pueden ver en la ilustración 19.

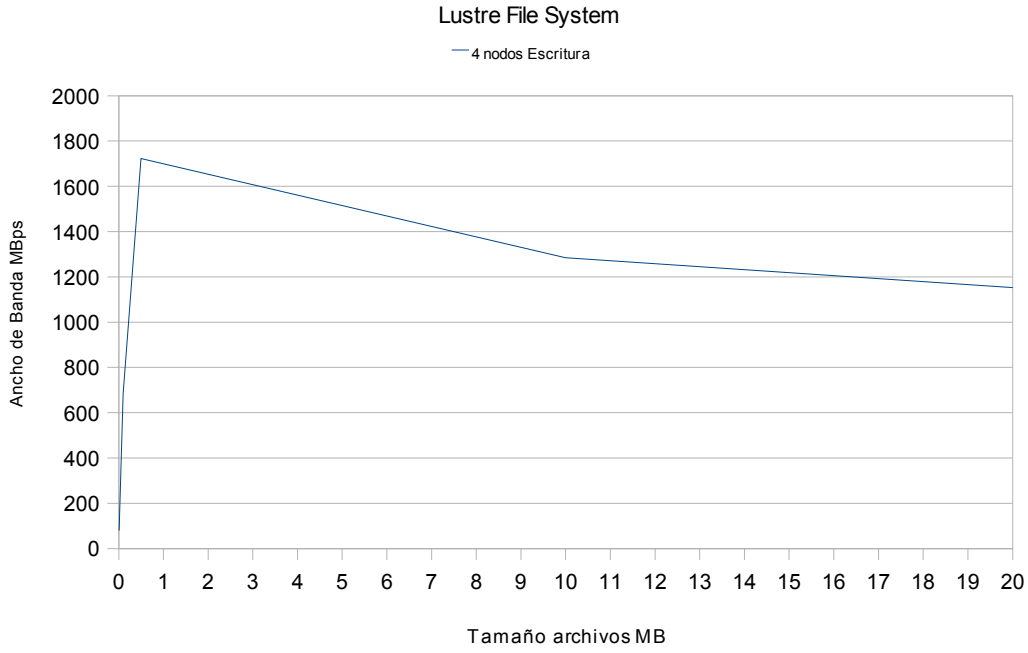


Ilustración 19 Escritura separada desde 4 nodos sobre 4 archivos separados usando el sistema de archivos Lustre.

Fuente: Autores

A diferencia de lustre PVFS2 siempre mantiene una tendencia creciente en el ancho de banda de escritura a medida que se aumentan los tamaños de los archivos que se escriben simultáneamente por los 4 nodos clientes del sistema de archivos, este comportamiento se puede observar en la ilustración 20.

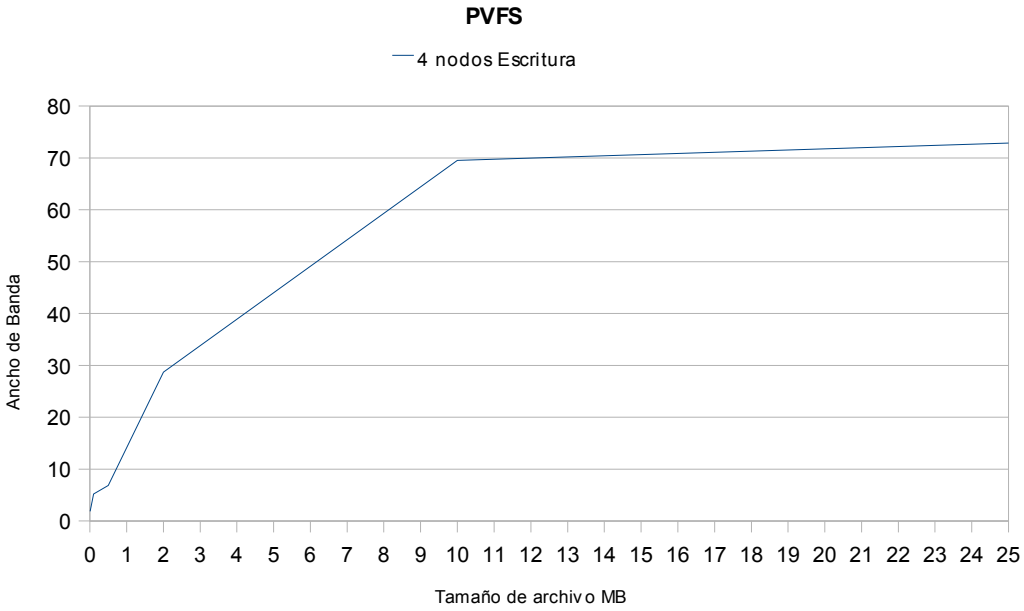


Ilustración 20 Escritura separada desde 4 nodos sobre 4 archivos separados usando el sistema de archivos Lustre.

Fuente: Autores

Nuevamente, Lustre presenta un mejor rendimiento que PVFS2 en el ancho de banda de escritura de archivos ahora de manera separada, la diferencia que se observa está en la utilización de archivos menores a 10 MB, las propiedades de Lustre sobre los clientes muestran un aporte significativo en el momento de hacer uso del sistema de archivos. Es claro que los dos sistemas de archivos tienden a estabilizarse cuando la infraestructura de red es saturada por el tamaño de la información que viaja por la misma, como se observa en la ilustración 21.

Ilustración 21 Comparación de lectura desde 4 nodos sobre 4 solo archivos entre PVFS2 y Lustre.
Fuente: Autores

Para el almacenamiento en una infraestructura diseñada, la densidad de los archivos que se manejan por el servidor de correo ascienden desde varios kilobytes hasta algunas megabytes, con este parámetro de uso se puede concluir que Lustre presentó el mejor rendimiento en los escenarios planteados para tamaños de archivos pequeños.

La arquitectura de Lustre refleja que tiene un mejor comportamiento en el acceso a los archivos almacenados desde el lado del cliente por medio de un conjunto de herramientas software como se muestra en la ilustración 22, la dinámica de funcionamiento ofrecida por Lustre permite tiempos de lectura mas bajos que los de PVFS2 como se observó en los escenarios probados.

Un factor que influye en el rendimiento de los sistemas de archivos es la limitación del tráfico de la información por la infraestructura de red usada, Lustre y PVFS2 mantiene en sus recomendaciones el uso de redes de alta velocidad como lo son Gigabit Ethernet, Myrinet o Infinitband entre otras, para ver el verdadero rendimiento que pueden ofrecer estas arquitecturas de sistemas de archivos paralelos.

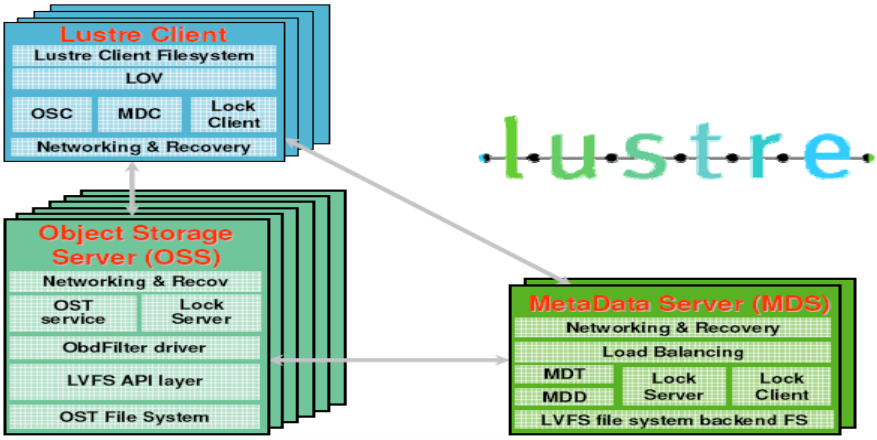


Ilustración 22: Componentes software de Lustre.

Fuente: HEPiX Storage Technology. http://hepix.caspur.it/storage/

7.4 COMUNICACIÓN ENTRE EL USUARIO Y EL SERVIDOR VIRTUAL

El método de reenvío de conexión que usa el cluster de alta disponibilidad es el de enrutamiento por encapsulado IP (VS-TUN), las conexiones ingresan por los nodos balanceadores quienes encapsulan los paquetes de solicitud dentro de paquetes IPIP para ser redirigidos a los servidores reales construyendo túneles virtuales IP, estos deben tener la capacidad de des-encapsular los paquetes IPIP[17] para ser procesados.

En cada unos de los servidores reales está configurada una interfaz virtual (tunl0) con la IP pública del servidor, interfaz por la que recibirán las peticiones del servicio. Las respuestas generadas son enviadas directamente a los clientes por

[17] http://en.wikipedia.org/wiki/IP_in_IP

los servidores sin necesidad de pasar por los nodos balanceadores, característica que evita los cuellos de botella en el sistema optimizando el uso de los canales de comunicación.

8. INTEGRACIÓN DE LOS COMPONENTES PARA EL DISEÑO DE ARQUITECTURA PLANTEADA

La arquitectura propuesta fue diseñada con la unión de componentes que aportan características de escalabilidad, rendimiento, y tolerancia a fallos mediante la replicación de servicios e información. Las tres capas que se muestran en la ilustración 23 anteriormente explicadas definen el orden en que una petición de servicio será atendida por el servidor virtual y su integración será descrita en esta sección.

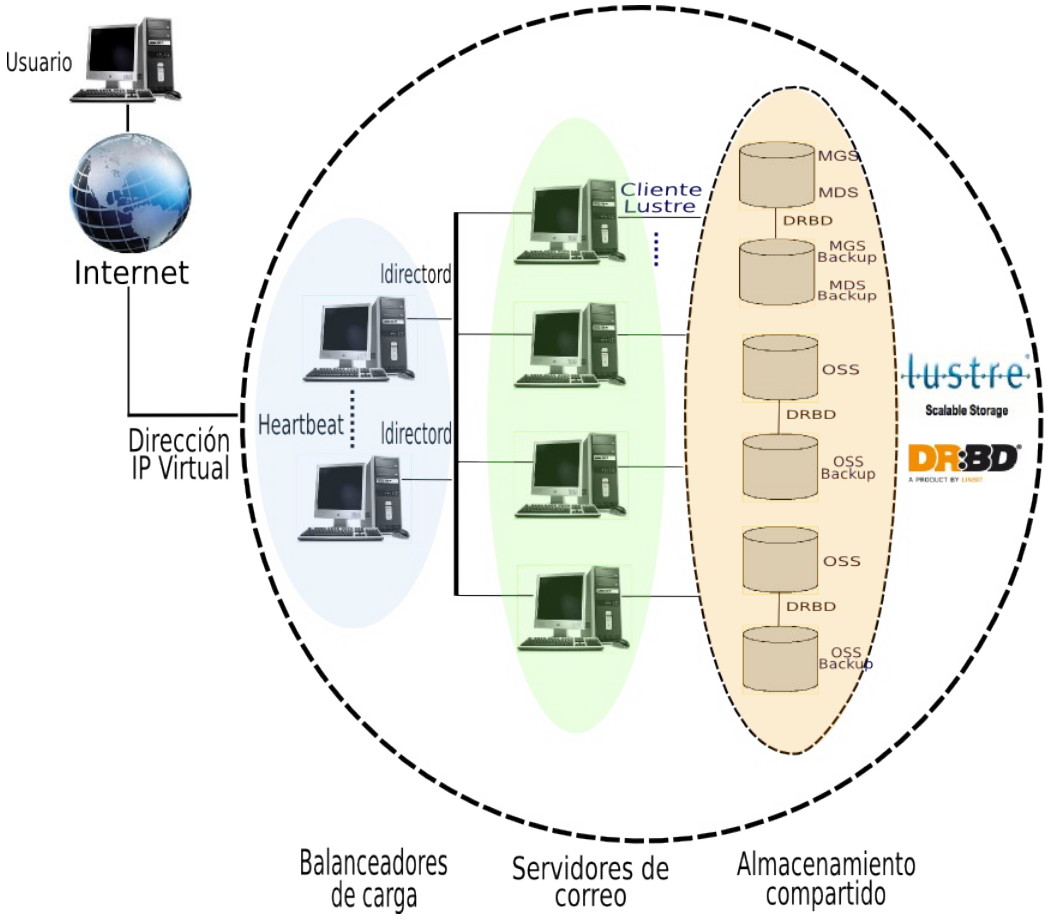


Ilustración 23: Arquitectura propuesta

Fuente: Autores

La primera de las capas es la encargada de recibir las peticiones de servicio y de

repartirlas en el conjunto de servidores reales. Su probabilidad de fallo se distribuye al estar compuesta por 2 computadores quienes permanecen en continua monitorización de los servicios prestados por el servidor virtual. La configuración de estos dos nodos es Activo/Pasivo, lo que significa que uno de los 2 estará como principal, procesando las peticiones de servicio de los usuarios y de otros servidores de correo que quieran establecer comunicación para el envío de mensajes. El otro nodo denominado secundario, estará esperando que el principal falle para retomar los servicios que se estaban prestando antes del suceso siendo este proceso transparente para el usuario. Los servicios monitorizados son 2, el primero es el de correo electrónico el cual funcionará por el puerto 25, el segundo es un servicio Web para la interfaz de acceso al servicio de correo y estará encaminado al puerto 80.

El software encargado de la monitorización de los servicios es heartbeat provisto por el proyecto Linux-HA. Con el demonio heartbeat se monta automáticamente la interfaz con la IP virtual del servidor, siendo ésta la IP por donde se reciben las peticiones de servicio por los puertos anteriormente mencionados.

Cuando el nodo principal falla, el demonio hearbeat será el encargado de levantar una interfaz con la IP virtual en el secundario y es por esta por donde se seguirán prestando los servicios del principal garantizando una continuidad en el funcionamiento del servidor.

En la ilustración 24 se muestra un pantallazo de un nodo balanceador establecido como primario el cual tiene la interfaz de red virtual donde está configurada la IP virtual del cluster (flecha roja en la ilustración 24), es por esta IP que se recibirán las peticiones de servicio de los usuarios y de otros SMTP en Internet. La flecha azul en la ilustración 24 muestra la interfaz real del nodo balanceador en la red interna del cluster.

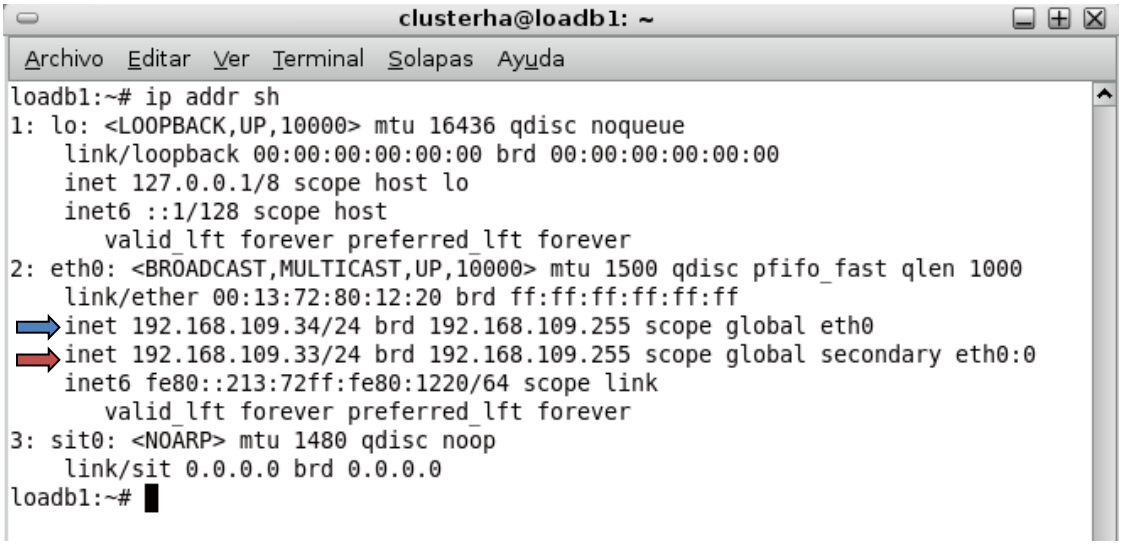


Ilustración 24: Interfaz de red del servidor Virtual

Fuente: Autores

En esta misma capa se lleva a cabo el balanceo de carga, tarea realizada por el demonio ldiretord el cual hace uso de reglas IPVSadm para actualizar la tabla de servidores reales disponibles para prestar servicio. El algoritmo de planificación para el balanceo de carga es el WLC (Servidor con menos conexiones activas ponderado), el cual permite asignar un peso a los servidores reales según sus capacidades o según las características de los servicios prestados. Este método garantiza una distribución de las peticiones de servicio según el estado de las conexiones activas en cada servidor evitando la saturación de los mismos. Para la implementación del algoritmo en el sistema de correo, se ha asignado el mismo valor de peso a los servidores reales por poseer las mismas características de hardware y de software.

En la ilustración 25 se muestra una captura de pantalla de la consola de uno de los balanceadores de carga donde se usa una regla ipvsadm (óvalo rojo) para listar los servidores reales y algunas estadísticas más. La lista se divide por servicios incluyendo la dirección IP virtual del cluster con el puerto para cada servicio que se está balanceando (óvalos azules). Debajo de la dirección IP virtual para cada servicio se encuentra la lista de los servidores reales disponibles para

redirigir peticiones de servicio.

Como se puede apreciar hay 2 servicios para ser balanceados, uno por el puerto 25 para comunicaciones con los SMTP externos y otro por el puerto 80 usado por los usuarios para acceder a la interfaz del servicio de correo a través de un Webmail. Las llaves verdes en la ilustración 25 indican el número de conexiones que el servidor virtual ha procesado; en total para el servicio de correo por el puerto 25 ha recibido 5 peticiones atendidas por 3 de los cuatro servidores reales, y para el servicio HTTP por el puerto 80 se han recibido 181 peticiones de servicio repartidas entre los servidores reales.

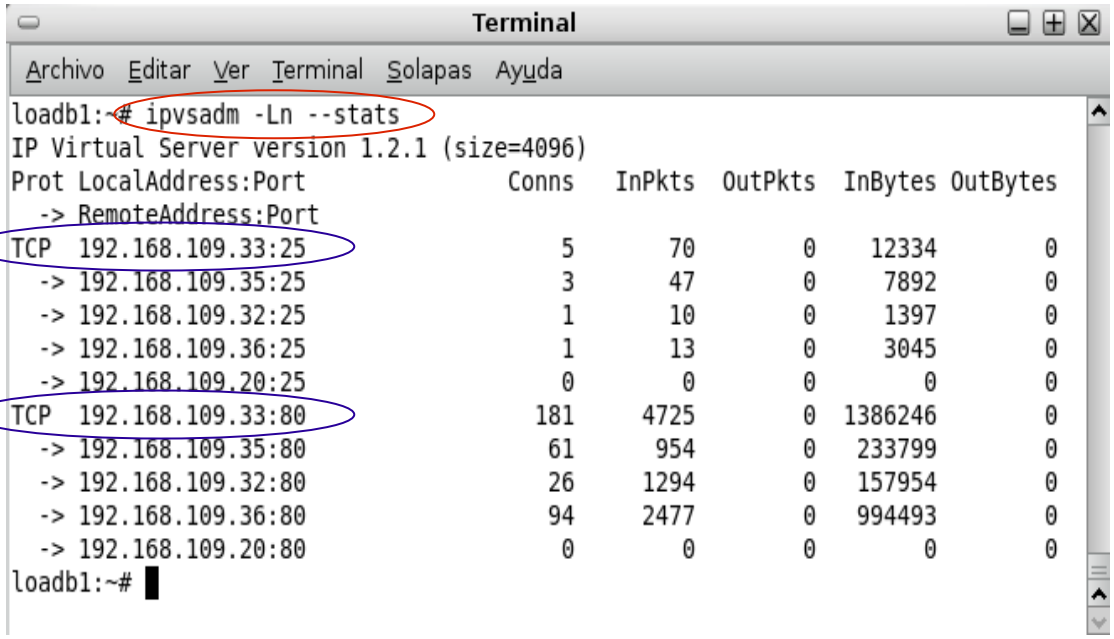


Ilustración 25: Ilustración 25: Tablas de servidores reales con reglas ipvsadm

Fuente: Autores

Dentro del diseño del cluster de alta disponibilidad, en la capa de servicio, se encuentran cuatro servidores reales que son los encargados de atender las peticiones del servicio después del proceso de balanceo de carga en el sistema. Cada unos de estos servidores reales tiene configurado el MTA Postfix para el envió y recepción de mensajes a través de Internet con la asignación de cinco

direcciones IP públicas proporcionadas por la División de Servicios de Información de la Universidad Industrial de Santander, de las cuales una es usada para la recepción de las solicitudes al servicio de correo por medio del protocolo SMTP por otros servidores de correo en Internet y las otras cuatro están asignadas a los servidores reales de correo para el envió de mensajes de correo fuera del cluster. El acceso a los buzones de correo es por medio del protocolo IMAP provisto por el servidor Courier, el cual se adapta a las características de escalabilidad necesarias en el cluster al ser capaz de administrar grandes cantidades de usuarios. La autenticación de los usuarios con el servidor IMAP es realizada por los mecanismos PAM y el método de autenticación es el clásico en los sistemas UNIX donde cada usuario posee un nombre de entrada al sistema o login y una clave o password.

El acceso al servicio de correo es por medio de la interfaz Web provista por *Roundcubemail*[18], el cual utiliza el protocolo IMAP para el acceso a los buzones de correo permitiendo a los usuarios acceder en cualquier lugar donde haya una conexión de Internet a través de un navegador. Este Webmail está realizado sobre php con ajax[19] (acrónimo de Asynchronous JavaScript And XML, JavaScript asíncrono y XML) brinda una interfaz más rápida y agradable para el usuario. Este software está disponible con licencias de código abierto, contando con gran respaldo en documentación por la comunidad de usuarios y de desarrolladores en el mundo.

El servidor Apache fue el utilizado para albergar el Webmail Roundcubemail, el cual está disponible a través de Internet por la asignación de las direcciones IP públicas provistas por la División de Servicios de Información de la Universidad Industrial de Santander. Así mismo, fue necesario la asignación de un DNS (*Domain Name System*), para el acceso más cómodo a la interfaz Web de los usuarios del servicio el cual tiene el nombre de: “http://*clustermail.uis.edu.co*”.

El servidor de correo cuenta con una interfaz entre el MTA Postfix y los paquetes software para la detección de virus y spam como se muestra en la ilustración 26.

[18] Proyecto Roundcubemail disponible en http://roundcube.net/

[19] http://es.wikipedia.org/wiki/AJAX

Por medio de la interfaz Amavis-new, mediante parámetros de configuración de Postfix se establecen 2 filtros cuyo funcionamiento está basado en la utilización de 2 herramientas las cuales son: SpamAssassin y ClamAV. Amavis-new controla el demonio encargado de filtrar el correo considerado como spam haciendo uso de algunas listas negras totalmente objetivas sobre la aceptación o no de un correo entrante. Si algún mensaje es catalogado como spam, este será descartado antes de llegar al usuario los cuales son descartados por el servidor SMTP. En el servidor de correo implementado no se permite la recepción de mensajes con archivos adjuntos de algunas extensiones como ejecutables (.exes, sh y otros más que se consideran software malicioso).

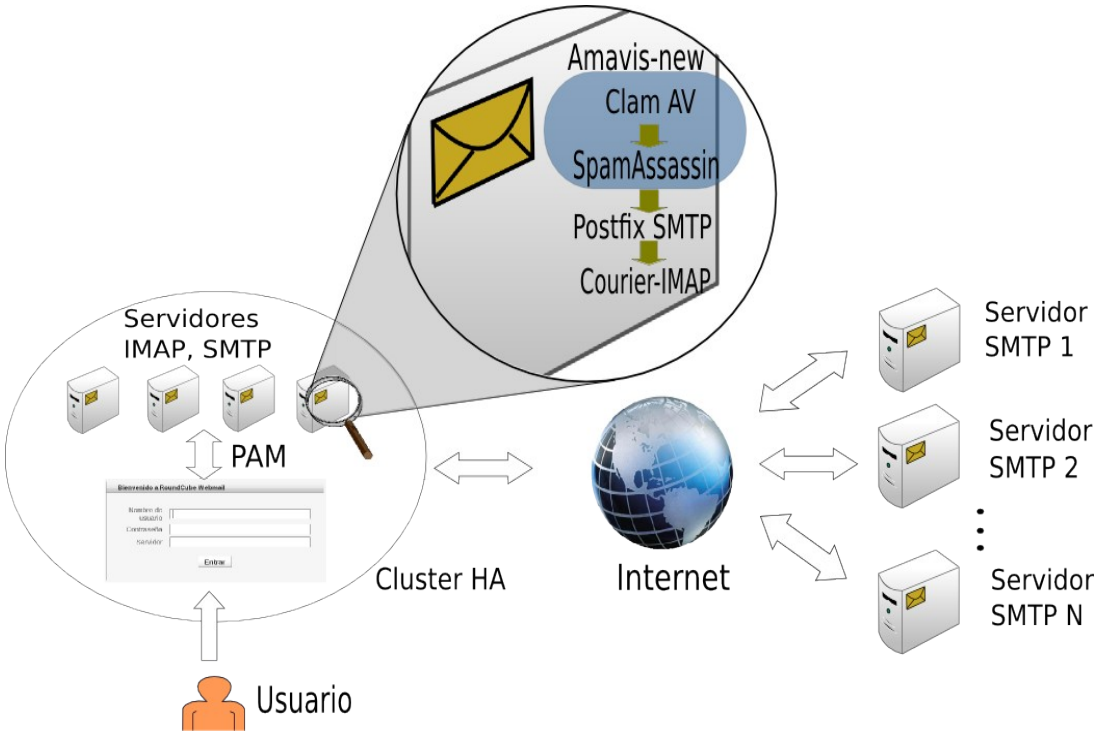


Ilustración 26: Esquema del servicio de correo en el Cluster HA

Fuente: autores

El sistema de archivos basado en cluster que se uso fue Lustre File System, el

cual se eligió basándose en los resultados obtenidos de las pruebas de rendimiento donde se comparó con PVFS2 (Parallel Virual File System 2), además de la revisión bibliográfica realizada donde se encuentra que Lustre tiene altos niveles de rendimiento y escalabilidad al lado de otros sistemas de archivos[19][21].

La distribución de los componentes del sistema de archivos en la arquitectura diseñada es como se muestra en la ilustración 23; un nodo de almacenamiento cumple con las labores de servidor de metadatos MDS, como también con las labores de administración del sistema de archivos con el sistema de gestión MGS. Existen 2 nodos servidores de almacenamiento (OSS en la ilustración 23) que aportan un volumen físico al sistema de archivos paralelo y al igual que el servidor administrador y de metadatos (MDS y MGS) tienen sus respectivas copias de respaldo por medio de DRBD. Esta integración permite obtener como resultado un sistema de archivos de alto rendimiento por medio del uso de paralelismo en la E/S, altamente escalable, y tolerante a fallos mediante la replicación en tiempo real.

El sistema operativo perteneciente el proyecto GNU/Linux usado en la implementación fue Debian en su rama estable, heartbeat y los paquetes necesarios para correr LVS en el cluster están disponibles en los repositorios oficiales, aunque es importante destacar que en cualquier sistema UNIX es posible configurar y construir sistemas distribuidos para prestar este tipo de servicios de red.

En la arquitectura está separado el servicio del almacenamiento, esto con la finalidad de disminuir la probabilidad de fallo. El sistema de archivos Lustre en su versión actual permite configurar nodos de respaldo, aunque por sí mismo no tiene la capacidad de replicación. Cuando algún servidor del cluster de almacenamiento falle, otro tomará su lugar, y permitirá transparencia en el servicio al usuario quien no notará el error. Este proceso debe ser automático, de tal manera que no se depende de un administrador que corrija los errores en el sistema.

La escalabilidad en el sistema está en todos los niveles. La versión actual de

heartbeat permite tener un cluster como sistema balanceador, permitiendo más capacidad de procesamiento y direccionamiento a una granja de servidores. En el cluster de almacenamiento se permite escalar hasta miles de nodos, característica propia del sistema de archivos paralelo Lustre. La granja de servidores reales puede escalar hasta 100 o más nodos, capacidad que esta ligada el método de enrutamiento del sistema balanceador.

9. FUNCIONAMIENTO DEL SERVICIO DE CORREO

El Webmail usado para la implementación del cluster de alta disponibilidad es Roundcubemail el cual tiene toda la funcionalidad de un cliente moderno de correo, incluyendo soporte MIME, lista de contactos, manejo de carpetas, búsqueda de mensajes y corrección ortográfica. A diferencia de otros clientes Web, su interfaz de usuario es muy parecida a una aplicación. Esto significa que tiene características como agarrar y soltar (drag-n-drop) usadas en aplicaciones normales de escritorio.

Para el ingreso al sistema de correo electrónico a través de Internet, se necesito solicitar un nombre de dominio (DNS) asociado a la IP pública del cluster provista por la División de Servicios de Información.

El nombre dominio asignado a la IP pública es: “clustermail.uis.edu.co” como se muestra en la ilustración 27. Este nombre de dominio se ingresa en el campo de URL de cualquier navegador de Internet, acción que muestra la *pantalla de entrada* (ilustración 27), con la cual se autentica el usuario.

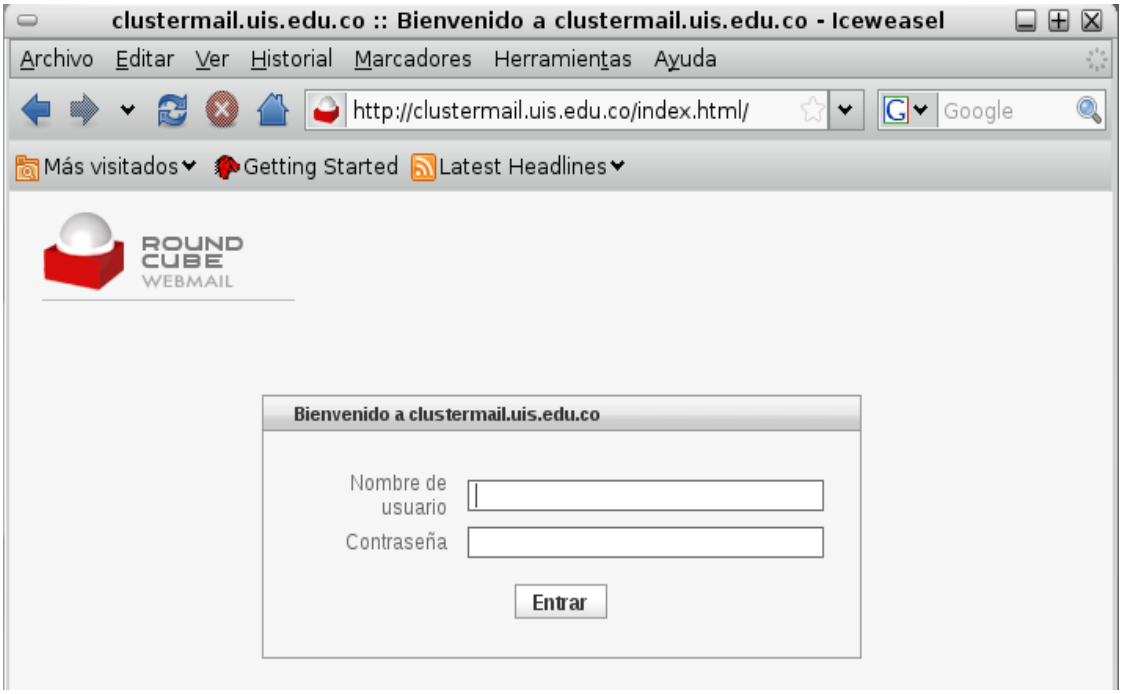


Ilustración 27: Pantalla de acceso al servicio de correo

Fuente: Autores

Después del ingreso del usuario en el sistema, se carga la interfaz para la visualización de los mensajes de correo, como también las carpetas por defecto en un cliente de correo para la clasificación de los mismos. Como se muestra en la ilustración 28, existen cuatro áreas principales las cuales se describen así:

- **Lista de carpetas** (1), en el menú a la izquierda de la pantalla se muestran todas las carpetas que tiene una cuenta de correo. Siempre se encontrará entre las cinco primeras: Entrada, Borradores, Enviados, Basura (Spam) y Papelera. Además de estas 5 carpetas es posible la creación de las que se desee. Alguna de las carpetas aparecen en negrilla y tienen a su derecha un número entre paréntesis. Esto indica que estas carpetas contienen mensajes no leídos y el número le indica cuantos. Para abrir una carpeta, se hace clic una vez en ella y la lista de mensajes aparecerá.
- **Barra de acciones** (2), Esta área de la pantalla contiene iconos que

permiten llevar a cabo diferentes acciones. Los 7 iconos tienen las siguientes funciones, de izquierda a derecha:

Refrescar Carpeta: Buscar nuevos mensajes en la carpeta actual.

Componer: Crear un nuevo mensaje de correo.

Responder: Crear un nuevo mensaje en respuesta al correo actualmente visible; se enviará exclusivamente a quien envió el mensaje seleccionado.

Responder a todos: Parecido a responder, pero adicionalmente se enviará la respuesta a todos los recipientes. Solamente tiene sentido si el mensaje fue enviado a un grupo de personas y usted desea que todas ellas reciban su respuesta.

Reenviar mensaje: reenvía el mensaje seleccionado a otra persona.

Eliminar: Eliminar el o los mensajes seleccionados, es decir, enviarlos a la Papelera.

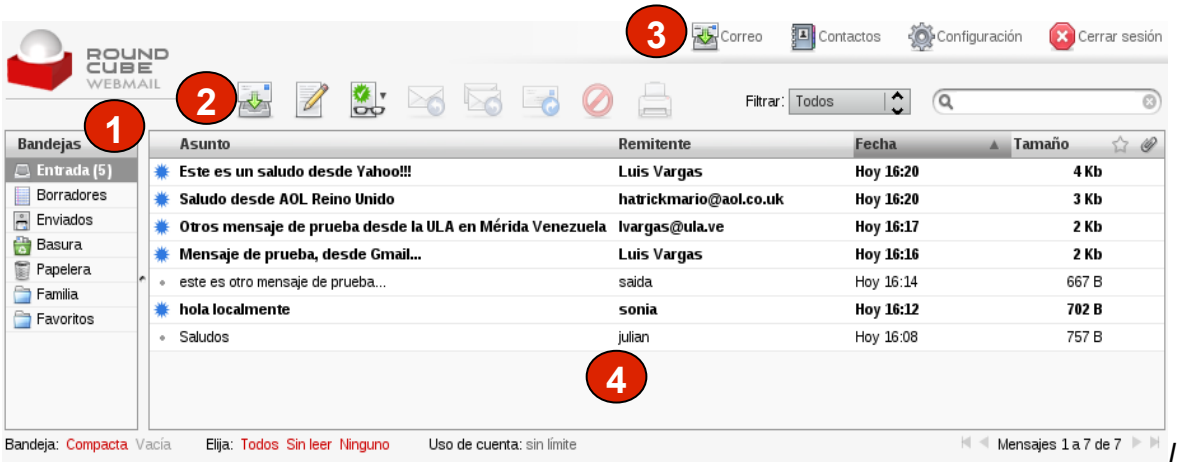
Imprimir: Imprimir el mensaje seleccionado.

A la derecha de la barra de acciones se puede ver el campo de búsqueda. Este campo le permite al usuario buscar en todos los mensajes en la carpeta actual, acción parecida a la búsqueda en un motor de búsqueda Web.

- **Barra de aplicación** (3), los 3 primeros iconos en la parte superior derecha de la pantalla le dan acceso a las aplicaciones que son parte de RoundCube. Estas incluyen el componente de correo que se ha estado describiendo. Adicionalmente está el libro de contactos que se describe posteriormente. El botón de configuración le da acceso a varias opciones para adecuar RoundCube según las necesidades. Finalmente se encuentra el botón de cerrar sesión que le permite salir de RoundCube. Es recomendable el uso del botón de Cerrar Sesión después de usar RoundCube para asegurar que nadie que use el mismo computador pueda acceder a sus correos electrónicos.
- **Lista de mensajes** (4), esta porción de la pantalla despliega la lista de

mensajes en la carpeta. Para ver un mensaje, se debe hacer doble-clic sobre el mismo. Se puede seleccionar también un mensaje haciendo clic una vez y después puede elegir una acción sobre este usando uno de los botones en la barra de acciones. Se puede seleccionar más de un mensaje manteniendo oprimida la tecla Ctrl y haciendo clic sobre varios mensajes uno tras otro. Finalmente, puede tomar y soltar los mensajes en otra carpeta. Esto ofrece una opción alternativa para eliminar mensajes descargándolos en la carpeta Papelera.

En la ilustración 28 se puede observar que algunos de los mensajes se indican con letra negrilla y una estrella azul a la izquierda de ellos. Estos mensajes son aquellos que no han sido leídos por el usuario. Un mensaje con un pequeño papel con clip a su izquierda indica que contiene un adjunto, tal como un documento PDF o una fotografía.



lustración 28: Interfaz de usuario principal

Fuente: Autores

La interfaz para escribir mensajes de correo se muestra en la ilustración 29, a continuación se describen las funciones más importantes.

- **Acciones sobre el mensaje** (1), las acciones que se pueden utilizar durante la realización de un mensaje de correo son: volver a la interfaz de usuario principal, enviar el mensaje a los destinatarios, corrección de ortografía, adjuntar archivo, y archivar como borrador.
- **Archivos adjuntos** (2), al hacer clic en el botón con el signo "+" también se puede adjuntar archivos, cuyo tamaño máximo es 5 MB.
- **Campos del mensaje de correo** (3), aquí se introducen los datos de los destinatarios como: dirección de correo electrónico, asunto del mensaje, la opciones para mensajes con copia oculta o visible, el contenido del mensaje y algunas opciones para la ortografía como también, botones para algunas acciones.

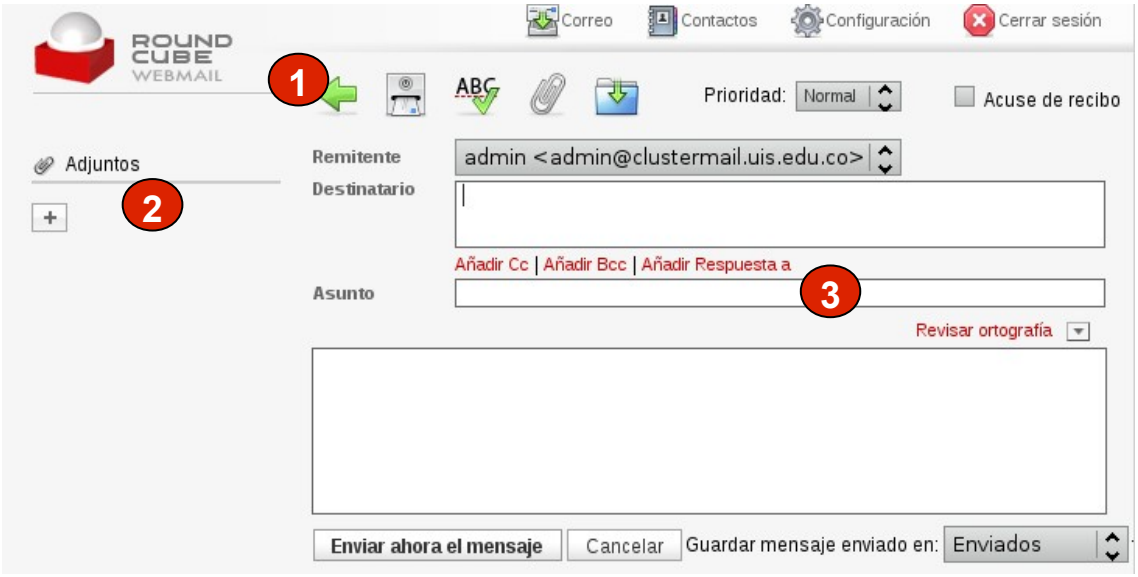


Ilustración 29: Interfaz para la creación de mensajes de correo

Fuente: Autores

10. PRUEBAS DE FUNCIONALIDAD DEL SERVICIO DE CORREO

Para la realización de las pruebas de funcionalidad se contó con la colaboración de los estudiantes pertenecientes a la Comunidad Universitaria de Software Libre CUSOL-UIS, personas estudiantes de otras carreras como también algunos estudiantes que realizan proyecto de grado en el Centro de Tecnologías y Comunicación CENTIC. Para tal fin, se crearon las respectivas cuentas de usuario y se les sugirió que utilizaran el servicio de correo dentro y fuera de la universidad durante la realización de las pruebas programadas cuya duración fue de una semana.

Se crearon en total 25 cuentas de usuario y se sugirió a la población de prueba que como mínimo se accediera 10 veces al día al servicio. También se sugirió la realización de envío y recepción mensajes de correo a través de otros servidores de correo disponibles en Internet como Gmail, Hotmail y Yahoo.

La encuesta realizada se encuentra en el anexo D y la tabla 1 muestra los resultados obtenidos de su aplicación.

Pregunta	SI	NO	Observaciones
1. ¿Cuando intentó acceder al servicio este estuvo disponible?	25	0	Servicio indisponible.
2. ¿Pudo realizar correctamente el envío de mensajes a los otros servidores de correo en Internet como Gmail, Hotmail y Yahoo?	24	1	En algunos casos el correo era considerado como spam.
3. ¿Pudo realizar correctamente la recepción de mensajes desde otros servidores de correo en Internet como Gmail, Hotmail y Yahoo?	23	2	En dos casos el mensaje tuvo que ser reenviado obteniendo respuesta correcta.
4. ¿Notó algún comportamiento anormal durante la utilización del servicio?	0	25	
5. ¿Le resultó práctica la interfaz Web del servicio de correo?	25	0	Agradable y con las mismas utilidades que otros servicios. Sugerencia sobre los botones para una indicación con texto además del icono.

Tabla 1: Resultado encuesta sobre la funcionalidad del servicio

La encuesta realizada permitió inferir aspectos importantes que se explican a continuación. Respecto a la pregunta número 1 se encontró que el 100% de los encuestados no tuvieron problemas con la disponibilidad del servicio.

Los resultados en la encuesta referentes a la pregunta número 2 reflejan que el 96% de los mensajes enviados desde el servicio implementado llegaron con éxito, y en 2 casos fue catalogado como spam. Lo anterior debido a problemas con el DNS de la Universidad y el proveedor del servicio de Internet Telecom. S.A, puesto que a pesar de tener los registros necesarios asignados por la División de Servicios de Información para la realización del proceso de búsqueda del DNS reverso[20] este no funciona correctamente.

La pregunta número 3 se refiere al proceso de recepción de correo electrónico enviado a través de otros servidores de correo en Internet al servidor virtual implementado en este proyecto. Según la encuesta, los mensajes entrantes al

[20] DNS reverso http://en.wikipedia.org/wiki/Reverse_DNS_lookup

servicio de correo desde otros servidores como Gmail, Hotmail y Yahoo fue satisfactorio al conseguirse un 92% de efectividad, salvo el caso de algunos usuarios quienes usaron servicios de correo sin centros de datos en Colombia como AOL del Reino Unido convirtiéndose en un inconveniente causado por la actual configuración de los servicios de red entre la Universidad y el ISP Telecom S.A.

La pregunta número 4 hace referencia al comportamiento del servicio durante su uso por parte de los usuarios, y en los resultados obtenidos encontramos que no se presentaron cierres inesperados ni otras situaciones en el servicio durante la realización de la prueba.

Para la mayoría de los usuarios de la población encuestada fue agradable el uso de una interfaz Web con las características de Roundcubemail por su funcionalidad y opciones disponibles para la realización de las tareas comunes en el servicio, aunque un 22% de los usuarios sugirieron que los botones para las acciones en la interfaz deberían tener su respectiva indicación en texto.

11. COMPARACIÓN DE ESCALABILIDAD ENTRE LA ARQUITECTURA IMPLEMENTADA Y UN SISTEMA CENTRALIZADO

Mediante la tabla 2 se pude ver la comparación entre un servidor convencional, que organiza todos los componentes del servicio de correo de forma centralizada, y una arquitectura distribuida como la implementada, es de resaltar que en la capa de servidores existe un cluster de computadores garantizando la redundancia en tarjetas de red, servicios de correo y sistema operativos, para llevar a cabo eventos de failover automáticamente como respuesta ante la presencia fallas en el servicio.

La capacidad de almacenamiento compartido en la arquitectura distribuida se realiza mediante el uso de Lustre File System y DRBD, los cuales proporcionan la unión de varios discos que conforman el almacenamiento total del cluster de alta disponibilidad.

Componente	Sistema Centralizado	Sistema Distribuido
# Tarjetas de red	1	4
# Servidores http, smtp, imap	1	4
Almacenamiento compartido	150 GB	518 GB

Tabla 2: Cantidad componentes en un sistema centralizado y un sistema distribuido.

Es importante que los servicios de red no estén sujetos a una capacidad de servicio limitada, una característica que resalta en el diseño de la arquitectura implementada es la capacidad de escalar frente a cambios en la demanda del servicio en la tabla 3 se muestra la escalabilidad que se obtiene en la arquitectura distribuida implementada haciendo uso del método de reenvío de conexión IP Tuneling[1].

El almacenamiento en el servicio de correo es proporcionado por Lustre File System el cual provee una escalabilidad de hasta 32 PetaBytes[27].

Componente	Sistema Centralizado	Sistema Distribuido
# Tarjetas de red	1-4	2-100
# Servidores http, smtp, imap	1-4	2-100
Almacenamiento comparido	150 GB-5 TeraBytes	32 PetaBystes

Tabla 3: Escalabilidad de los componentes de un sistema centralizado y un sistema distribuido.

Cuando una organización se enfrenta ante cambios en la demanda del servicio puede aumentar la capacidad computacional y de almacenamiento del servidor que presta dicho servicio, este incremento se puede hacer aumentando en 1, 2, hasta 5 veces los recursos que se tenían, pero cuando la demanda de servicio necesita que se hagan aumentos de 10, 50, 100 veces en los recursos que se tenían. Estos aumentos significativos no se pueden hacer en una arquitectura centralizada, mientras que con un buen diseño en una arquitectura distribuida se podría escalar mucho más que lo que se permite en un servidor centralizado en capacidad de cómputo y de almacenamiento como se ve reflejado en la arquitectura implementada.

12. PRUEBAS DE FAILOVER EN EL FUNCIONAMIENTO DEL SERVICIO DE CORREO

Las pruebas de failover se realizaron apagando los equipos y midiendo el tiempo que se tardaba en retomar los servicios al nodo que tenía replicado el servicio adjunto. Este proceso se realizó durante 8 veces y se promediaron los resultados. Primero se realizó la prueba en la capa de balanceo de carga, en esta se apagaba el nodo que aceptaba en esos momentos las solicitudes de servicio (activo) y era retomada por el otro nodo (pasivo). El promedio del tiempo de retoma de servicios en la capa de balanceo de carga fue de 4,351 segundos, los resultados de esta prueba se muestran en la ilustración 30.

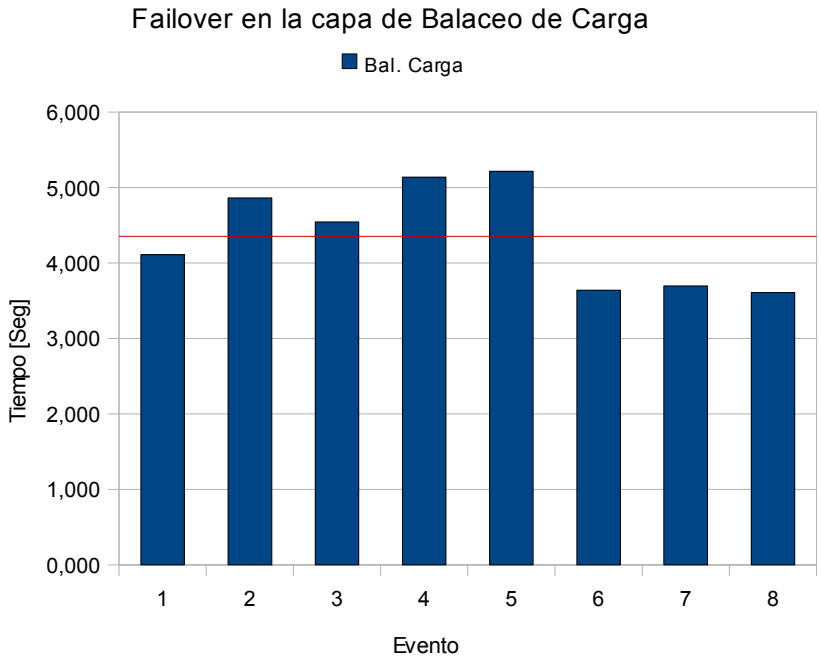


Ilustración 30 Tiempo de retoma de servicio en la capa de balanceo de carga

Fuente: Autores

En la capa de cluster de servidores existen múltiples nodos activos que están disponibles siempre para atender las necesidades de servicio, el fallo de un nodo

solamente conlleva una reconexión con otro servidor del cluster. Es de esperarse que este tiempo sea muy pequeño debido a que existen otros servidores activos para atender el servicio. En promedio el tiempo de reconexión en la capa de cluster de servidores fue de 0,762 segundos como se ve en la ilustración 31 en la línea de color rojo.

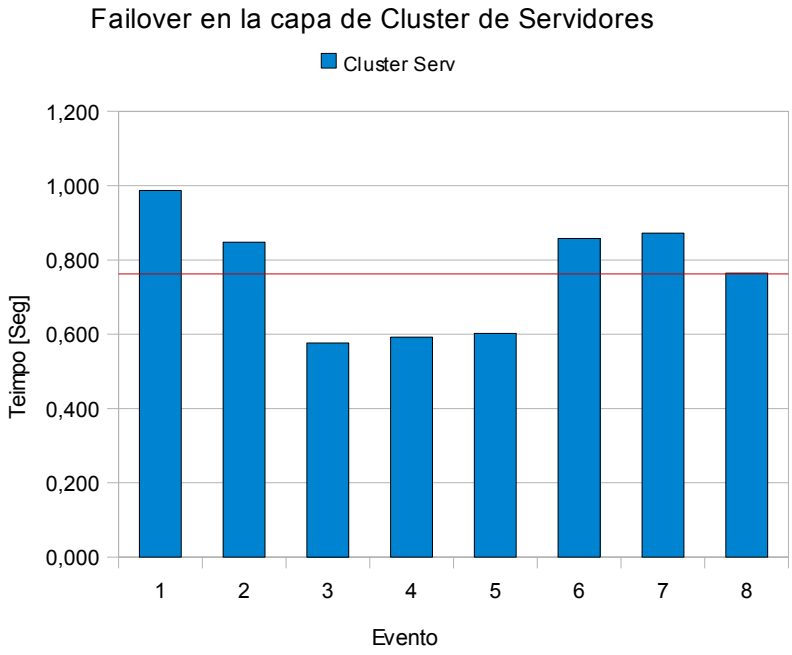


Ilustración 31 Tiempo de retoma de servicio en la capa de servidores reales

Fuente: Autores

Cuando se realizó la prueba en el sistema de archivos paralelo tolerante a fallos, el tiempo de reconfiguraron del sistema de archivos realizado por el MGS de Lustre FS para incluir la copia proporcionada por DRBD, mostró el tiempo mas alto de failover en todas las capas, este obtuvo un promedio de 137,162 segundos los resultados se muestran en la ilustración 32.

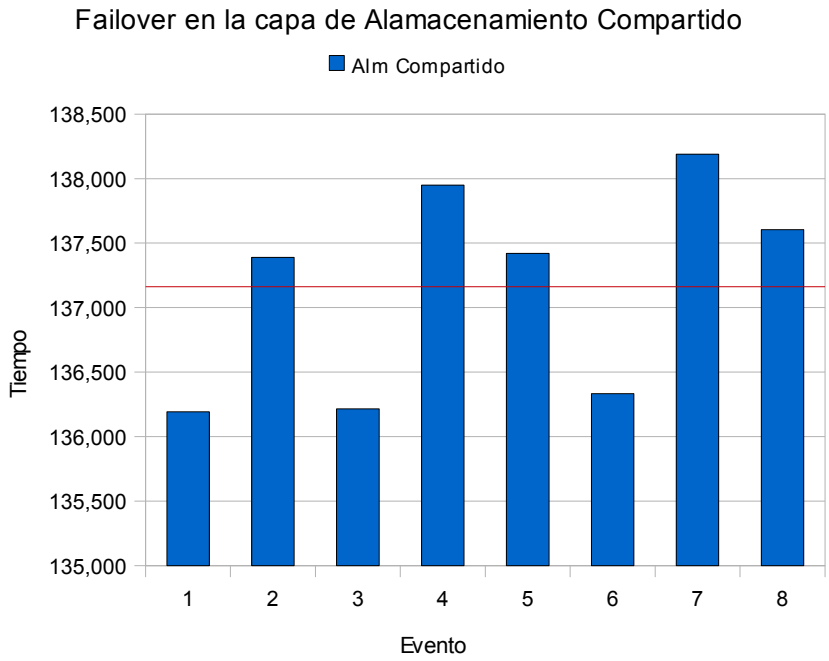


Ilustración 32 Tiempo de retoma de servicio en la capa almacenamiento compartido

Fuente: Autores

El proceso de retoma de servicios se llevo a cabo con éxito en todos los componentes replicados del cluster de alta disponibilidad, todas las retomas de servicio o reconexiones se realizan automáticamente por medio de heartbeat, en el caso del balanceo de carga y las copias de respaldo en el sistema de archivos, y ldirectord administrando el número de servidores reales disponible para el reenvío de solicitud del servicio.

Las pruebas de failover realizadas mostraron que el proceso de retoma de servicio se lleva a cabo con tiempos diferentes, en la capa de almacenamiento compartido reflejo un tiempo superior a 2 minutos usados en la reconfiguración del sistema de archivos Lustre FS, mientras que la capa de cluster de servidores reflejo un tiempo de alrededor de 1 segundo pues solamente hay un proceso de reconexión.

13. CÁLCULO DE LA DISPONIBILIDAD

13.1 CÁLCULO DE LA DISPONIBILIDAD EN LA CAPA DE LOS NODOS BALANCEADORES

En términos de disponibilidad en los nodos de balanceo de carga, se presenta un cluster organizado de forma Activo/Pasivo, como se muestra en la ilustración 33, una máquina se encuentra activamente recibiendo las solicitudes de peticiones a la IP Virtual cuando y cuando esta falle la máquina que está como pasivo cambiara como activo y retomara las solicitudes que se realizan sobre la IP Virtual.

Activo / Pasivo

Ilustración 33: Organización Activo/Pasivo

Fuente: Autores

El cálculo de la disponibilidad se basa en las ecuaciones expuestas en la sección 6.9 del presente libro. Para calcular la disponibilidad se tienen los siguientes datos:

n = 2 nodos

a = 0,999118607[21]

$MTFO_{Balanccarga}$ = 4,351 Seg[22]

$MTBF$ = 8640 horas[23]

La variable c de la ecuación $F_d = c * (1-a)^n$, la cual define el número de formas en las que pueden fallar los componentes del cluster para la configuración Activo/Pasivo usada en los nodos balanceadores, c está definida por el número de combinaciones de todas las formas de daños que puedan existir y esta dado por $n * (n-1)$, pero se ha contado 2 veces las combinaciones en las que el servicio

[21] Valor de la disponibilidad como resultado de cálculo realizado a una máquina Dell Optiplex gx620 con sistema operativo GNU/Linux Debian 4.0

[22] Valores calculados en la sección 10 del presente libro

[23] Valor suministrado por el CENTIC para máquinas Dell Optiplex gx620

prestado por los nodos balanceadores está caído, se define entonces de la siguiente manera $c = \frac{n*(n-1)}{2}$ [23].

Con estos parámetros definidos las ecuaciones (4) (5) (6) (7) de la sección 6.9 del presente libro.

$F_d = 7{,}768536204\text{e-}7$

$F_t = 2{,}797710905\text{e-}7$

$F = 1{,}056624711\text{e-}6$

$D_{Bal\ carga} = 9{,}999989434\text{e-}1$

La disponibilidad de 5 nueves obtenida en la capa de balanceo de carga muestra que la recepción de la IP virtual en un año estará no disponible menos de 5 minutos, este valor refleja un buen comportamiento en términos de disponibilidad para esta capa[26].

13.2 CÁLCULO DE LA DISPONIBILIDAD EN LA CAPA DE SERVIDORES REALES

En la parte de cluster de servidores en términos de disponibilidad la organización es diferente, todos los servidores se encuentran Activos, como se muestra en al ilustración 34, cuando un servidor presenta fallas simplemente es retirado de la lista de los servidores disponibles para dar respuesta a las solicitudes reenviadas por los balanceadores de carga. La probabilidad de que falle el cluster está dada por la probabilidad de que falle un nodo dado que ya fallaron los otros tres servidores, a mayor número de nodos de servicio este valor tiende a cero.

Ilustración 34: Organización de un cluster de nodos activos

Fuente: Autores

El cálculo de la disponibilidad se basa en las ecuaciones expuestas en la sección 6.9 del presente libro. Para calcular la disponibilidad se tienen los siguientes datos:

n = 4 nodos

a = 0,999118607

$MTFO_{ClusterServ}$ = 0,762 Seg

$MTBF$ = 8640 horas

La variable c de la ecuación $F_d = c * (1 - a)^n$, la cual define el número de formas en las que pueden fallar los componentes del cluster de nodos de servicio Activos, c está definida por el número de combinaciones de todas las formas de daños que puedan existir sin repetir, solo existe una configuración que es la que conforman los 4 servidores en este caso $c = 1$.

La ecuación $F_f = n \frac{MTFO}{MTBF}$, tiene algunos cambios al no presentar failover por cada nodo que pertenece al cluster de servidores de esta forma $F_f = \frac{MTFO}{MTBF}$ para sistemas en los cuales todos los nodos son Activos.

Con estos parámetros definidos las ecuaciones (4) (6) (7) y la anterior modificada $F_f = \frac{MTFO}{MTBF}$.

$F_d = 6{,}035015476\text{e-}13$

$F_t = 2{,}449845679\text{e-}8$

$F = 2{,}449906029\text{e-}8$

$D_{Nodos\,de\,Serv}$=9,999999755e-1

Mediante la replicación del servicio de correo en cuatro nodos y mantener los cuatro nodos activos, se presentan tiempos de failover muy pequeños de menos de 1 segundo, garantizando que el servicio de correo es listo para ser usado por casi todo el año con tan sólo un tiempo de no funcionamiento de algunos segundos, valor de disponibilidad que es muy bueno y muy difícil de obtener[26].

13.3 CÁLCULO DE LA DISPONIBILIDAD EN LA CAPA DE ALMACENAMIENTO COMPARTIDO

La organización que se realizó en los componentes del almacenamiento usando Lustre, es un cluster de nodos Activo/Pasivo. Cada componente del sistema de archivos Lustre (MDT, OSS) se encuentran replicados por medio de DRBD, como se muestra en al ilustración 35. El proceso de failover tiene una duración de 200, este lo realiza la herramienta heartbeat provista por le proyecto Linux-HA, este tiempo de penalización es usado por el sistema de archivos Lustre para reconfigurar los elementos con los cuales cuenta para realizar el almacenamiento de la información.

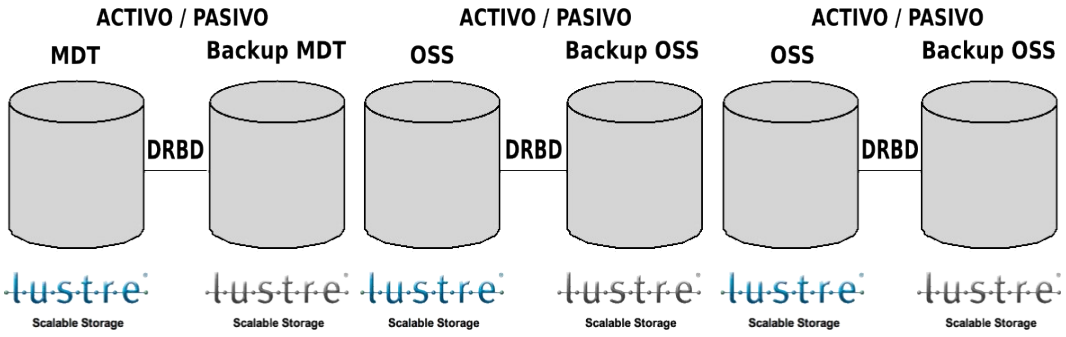


Ilustración 35: Organización Activo/Pasivo del sistema de archivos Lustre

Fuente: Autores

El cálculo de la disponibilidad se basa en las ecuaciones expuestas en la sección

6.9 del presente libro. Para calcular la disponibilidad se tienen los siguientes datos:

n = 6 nodos

a = 0,999118607

$MTFO_{Almaccompartido}$ = 137,162 Seg

$MTBF$ = 8640 horas

En el sistema de archivos Lustre replicado, si dos nodos fallan el sistema de archivos no funcionara, la variable c muestra que si hay un cluster de n nodos existe $\frac{n*(n-1)}{2}$ combinaciones únicas de un fallo doble de nodos.

Con estos parámetros definidos las ecuaciones (4) (5) (6) (7).

$F_d = 1{,}165280431\text{e-}5$

$F_t = 2{,}645871914\text{e-}5$

$F = 3{,}811152345\text{e-}5$

$D_{Almaccomp} = 9{,}999618885\text{e-}1$

El valor mas alto de failover que se observa esta en el almacenamiento compartido, este valor de aproximadamente 140 segundos la disponibilidad del sistema de archivos paralelo tolerante a fallos, el valor de disponibilidad obtenido da como resultado un no funcionamiento de menos de 1 hora al año, valor de disponibilidad bueno como resultado de unir Lustre y DRBD.

13.4 RESULTADO DE LA DISPONIBILIDAD DE LA ARQUITECTURA PROPUESTA

Teniendo en cuenta la ecuación (1) enunciada en la sección 6.9 y reemplazando los valores hallados se tiene:

$D_{Total} = D_{Balanceodecarga} * D_{Nodosdeservicio} * D_{Almacenamientocompartido}$ (1)

$D_{Total} = 9{,}999989434\text{e-}1 * 9{,}999999755\text{e-}1 * 9{,}999618885\text{e-}1$

$D_{Total} = 9{,}999608074\text{e-}1$

$Disponibilidad_{Total\%} = 99{,}99608\ \%$

Con la arquitectura diseñada e implementada se logró una disponibilidad de cuatro nueves, se puede decir que a lo largo de un periodo de un año el sistema estará fuera de servicio por 20 minutos 36 segundos. Valores de disponibilidad mayores a 99% son considerados sistemas de alta disponibilidad, y al alcanzar un valor de cuatro nueves la arquitectura propuesta es aceptada para prestar servicios de misión crítica[26]. Es de resaltar la redundancia como característica principal en cada una de las capas mejorando la disponibilidad de cada una de ellas distribuyendo la probabilidad de fallos en las copias activas o pasivas existentes en cada capa de la arquitectura implementada, la mejor manera de organizar los componentes de un cluster en términos de disponibilidad fue la usada en el cluster de nodos de servicios, donde todos los nodos se encuentran activos. El tiempo de failover es la variable que mas impacto causa sobre la arquitectura, es importante optimizar ese valor si se desea lograr índices mas altos de disponibilidad.

En aplicaciones de e-commerce es necesario que las arquitecturas encargadas de prestar estos servicios garanticen una disponibilidad mayor a 4 nueves (>99,99)[26], la disponibilidad lograda en la arquitectura propuesta esta en la capacidad de atender este tipo de servicios en los cuales el tiempo de no funcionamiento representan perdidas de dinero como lo revelan los estudios realizados por Giga Group[24] y Eagle Rock Alliance,Ltd[25].

[24] Firma de analistas internacional en tecnología de información. Resultados del estudio se encuentran disponibles en: http://www.dba-oracle.com/art_dbazine_high_avail.htm.

[25] Eagle Rock Alliance, Ltd es una empresa consultora especializada en administración de negocios, en la planificación de Fiabilidad de consultoría y tecnología. Resultados del estudio se encuentran disponibles en: http://www.beechtek.net/page/1012471

14. CONCLUSIONES

Los objetivos planteados al inicio del proyecto se han cumplido, el cluster de alta disponibilidad es funcional y cumple con los requisitos de escalabilidad y fiabilidad esperados. La elección de las herramientas seccionadas estuvo basada en una revisión bibliográfica donde se tuvieron en cuenta aspectos como la integración con los componentes de la arquitectura, las licencias de uso, además de las pruebas de rendimiento para la elección del sistema de archivos tolerante a fallos.

Por medio de un correcto diseño e integración de software libre y hardware convencional se puede construir una arquitectura distribuida tipo cluster, logrando alcanzar un índice de disponibilidad aproximado de 99,996% en el servicio de correo electrónico. La organización de los componentes en el diseño por capas permitió mantener definidos, separados y replicados los elementos del cluster de alta disponibilidad, distribuyendo la probabilidad de fallos entre las capas de la arquitectura.

La configuración en términos de disponibilidad en la arquitectura implementada se realizó con una configuración de activo/pasivo para las capas de balanceo de carga y almacenamiento compartido, diferente a la realizada en la capa del cluster de servidores donde se tiene una configuración de múltiples activos. Dentro de la realización de las pruebas para los escenarios creados con eventos de failover, se observó el impacto del tiempo de retoma de servicios sobre el índice de disponibilidad total del sistema, donde el tiempo más alto de failover se presentó en la capa de almacenamiento compartido.

Las pruebas realizadas sobre los sistemas de archivos permitieron elegir a Lustre File System el cual reflejó el mejor comportamiento en el acceso a los archivos almacenados desde el lado del cliente, esto es debido al conjunto de propiedades que permiten optimizar su uso. Con la unión de Lustre File System y DRBD se logró obtener un sistema de archivos con alto rendimiento en la E/S y al mismo tiempo con propiedades de tolerancia a fallos por medio de la replicación en

tiempo real.

El estado actual de las redes de conexión de la Universidad no favorecen un escenario de total aprovechamiento de la infraestructura del cluster de alta disponibilidad para la prestación de servicio de correo, por tal razon el comportamiento del sistema se ve afectado por los componentes de comunicación (interfaces y medios), que determinan los tiempos de trasmisión de datos y respuestas siendo un factor que influye en el desempeño general del cluster.

15. RECOMENDACIONES

Se debe garantizar el uso de técnicas de replicación de componentes en este tipo de arquitecturas que permitan la distribución de la probabilidad de fallo y estas deben estar presentes en todas las capas del sistema. Las decisiones en la organización deben estar basadas en un análisis de requerimientos del servicio a prestar donde se deben tener en cuenta aspectos como la demanda esperada de servicio y la relación costo/beneficio de su implementación. Se recomienda la realización de un estudio detallado de las características del servicio a prestar que permita tomar las mejores decisiones para el correcto funcionamiento del sistema en la organización.

La infraestructura de red usada para la implementación del cluster de alta disponibilidad no es especializada ni dedicada, factores que afectan la comunicación interna del cluster como también la comunicación con el usuario, esto debido a la existencia de canales de transmisión de datos limitados y compartidos. Se recomienda el uso de una infraestructura de comunicaciones que haga mejor uso de las características de los sistemas distribuidos al aumentar los canales de transmisión de datos mejorando el desempeño total del sistema.

La seguridad en este tipo de sistemas basados en cluster requiere de políticas a nivel de comunicación con el usuario, como también al interior de la arquitectura entre cada capa. Debido a esto se recomienda hacer uso de mecanismos como el empleo de credenciales y firewalls para la autorización en el acceso al cluster, métodos de autenticación de usuarios para el acceso al servicio, y el uso de técnicas de cifrado para la comunicación interna entre los componentes del sistema.

Cuando se necesita extender los niveles de disponibilidad en arquitecturas distribuidas para que estén en la capacidad de mantener los servicios y sus aplicaciones, aún cuando todo un lugar sea inhabilitado por un desastre (natural, eléctrico, etc.), se recomienda la distribución geográfica de los diversos componentes del cluster de alta disponibilidad proporcionando copias de respaldo

ante los diversos eventos de daños locales o regionales que se presenten.

16. REFERENCIAS

[1] Zhang W. Linux Virtual Servers for Scalable Network Services, Linux Symposium (July. 2000) DOI=http://www.linuxvirtualserver.org/ols/lvs.ps.gz.

[2] Yasushi Saito, Brian Bershad, Hank Levy. Manageability, Availability and Performance in Porcupine: a Highly Scalable, Cluster-Based Mail Service. 17th Symposium on Operating Systems Principles (SOSP), Diciembre; recibió **best paper** award. Disponible en: ACM Transactions on Computer Systems, August 2000.

[3] Pablo Galdámez Saiz. Hidra: Una Arquitectura para Alta Disponibilidad en Sistemas Distribuidos. Soporte a Objetos. Tesis doctoral departamento de sistemas informáticos y computación, Universidad Politécnica de Valencia. Pag 4,5. 2001.

[4] Fox, A., Gribble, S. D., Chawathe, Y., Brewer, E. A., and Gauthier, P. 1997. Cluster-based scalable network services. SIGOPS Oper. Syst. Rev. 31, 5 (Dec. 1997), 78-91. DOI= http://doi.acm.org/10.1145/269005.266662

[5] E. Marcus y H. Stern Blueprints for availability. Capítulo 5, págs. 78-79. John Wiley and Sons, New York (2003).

[6] M. D. Schroeder. A state-of-the-art distributed system: Computing with bob. En S. J.Mullender, editor, Distributed Systems, capítulo 7, págs. 1–16. Addison-Wesley, Wokingham, UK, 2nd edición, 1993. (P).

[7] Doreen L. Galli. Distributed Operating Systems: Concepts and Practice. Prentice Hall, Inc,Upper Saddle River, New Jersey, USA, Marzo 1993.(P)

[8] María C. España Boquera. Ediciones Díaz de Santos, Servicios avanzados de telecomunicación. Publicado en 2003. Capitulo 6, Sección 3. (P)

[9] E. Levy, A. Silberschatz Distributed file systems: concepts and examples ACM Computing Surveys (CSUR) Volume 22, Issue 4 (December 1990) Pages: 321 – 374 New York, NY, USA 1990. (P).

[10] J. P. Jesan. Information Security Doctoral Student Graduate School of Computer Information Sciences. http://www.acm.org/ubiquity/views/v7i02_InfoSecurity.html?searchterm=integrity+in+the+information 2008(P)

[11] R. Sandberg, D. Goldberg, S. Kleiman, D Walsh, and B. Lyon. Design and implementation of the SUN network filesystem. In P roceedings of the 1985 USENIX Conference, 1985. (P)

[12] M. Satyanarayanan, James J. Kistler, Puneet Kumar, Maria E. Okasaki, Ellen H. Siegel, and David C. Steere. Coda: A highly available file system for a distributed workstation environment. I EEE Transactions on Computers, 39(4):447–459, 1990.(P)

[13] Steven R. Soltis, Thomas M. Ruwart, and Matthew T. O'Keefe. The Global File System. In Proceedings of the Fifth NASA Goddard Space Flight Center Conference on Mass Storage Systems and Technologies, September 1996. (P)

[14] Y.N. Patt. The I/O subsystem: A candidate for improvement. 27(3):15–16, March 1994(P)

[15] D. A. Patterson, G. Gibson, and R. H. Katz. A case for Redundant Arrays of Inexpensive Disks (RAID). In P roceedings of ACM SIGMOD, pages 109–116, June 1988. (P)

[16] Park and K. Balasubramanian. Providing fault tolerance in parallel secondary storage systems. Technical Report CS-TR-057-86, Department of Computer Science, Princeton University, November 1986.(P)

[17] P. Carns Walter, R. B. Ross Rajeev PVFS: A Parallel File System for Linux Clusters Parallel Architecture Research Laboratory Clemson University, Clemson. 2000(P)

[18] ROMIO, MPI IO interface http://www.mcs.anl.gov/romio

[19] Cluster File Systems Inc. Lustre file system. http://manual.lustre.org/manual/LustreManual16_HTML/index.html

[20] F. Schmuck and R. Haskin GPFS: A Shared-Disk File System for Large Computing Clusters IBM Almaden Research Center, Proceedings of the Conference on File and Storage Technologies 2002 Monterey

[21] Sebepou, Z., Magoutis, K., Marazakis, M., and Bilas, A. 2008. A comparative experimental study of parallel file systems for large-scale data processing. In First USENIX Workshop on Large-Scale Computing (Boston, MA, June 22 - 27, 2008). USENIX Association, Berkeley, CA, 1-10.

[22] Proyecto Desarrollo IMAP DOI= http://www.imap.org/about/whatisIMAP.html

[23] Calculating Availability - Cluster Availability, Availability Digest, (May 2007) DOI=http://www.availabilitydigest.com/private/0205/calculating_availability_clusters.pdf

[24] Robinson R. y Polozoff A. Planning for Availability in the Enterprise, IBM WebSphere Developer Technical Journal (Dec 2003) DOI=http://www.ibm.com/developerworks/Websphere/techjournal/0312_polozoff/polozoff.html

[25] Diana Mullet, Kevin Mullet. Managing IMAP, O'Reilly 2000 Capítulo 1: The Internet Mail Model.

[26] E. Marcus y H. Stern Blueprints for availability. capítulo 2, págs. 9–30. John Wiley and Sons, New York (2003).

[27] D. Sanchez Design of store-and-forward servers for digital media distribution University of Amsterdan (Aug 2007).

[28] Diana Mullet, Kevin Mullet. Managing IMAP, O'Reilly 2000 Capítulo 2: What its IMAP? IMAP and POP A Comparison.

17. ANEXOS

Anexo A, Archivos de configuración en la capa de balanceo de carga

- ha.cf

```
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility     local0
keepalive 2
deadtime 10
initdead 120
udpport 694
bcast   eth0
auto_failback off
node    loadb1
node    loadb2
ping 192.168.109.1
respawn hacluster /usr/lib/heartbeat/ipfail
apiauth ipfail uid=hacluster gid=haclient
```

- haresources

```
loadb1 IPaddr2::192.168.109.33/24/eth0/192.168.109.255 \
directord::ldirectord.cf LVSSyncDaemonSwap::master
```

- authkeys

```
auth 2
2 sha1 clusterha
```

- ldirectord.cf

```
checktimeout=10
checkinterval=2
autoreload=no
logfile="/var/log/ldirectord.log"
quiescent=no
# IPVirtual para servidor http 80
virtual=192.168.109.33:80
        real=192.168.109.20:80 ipip
        real=192.168.109.36:80 ipip
        real=192.168.109.32:80 ipip
        real=192.168.109.35:80 ipip
         service=http
         request="solicitud.html"
         receive="test page"
         scheduler=wlc
         persistent=120
         protocol=tcp
         checktype=negotiate #ahi q probar con y sin esta opcion
         netmask=255.255.255.255
# IPVirtual para servidor SMTP 25
virtual=192.168.109.33:25
        real=192.168.109.20:25 ipip
        real=192.168.109.36:25 ipip
        real=192.168.109.32:25 ipip
        real=192.168.109.35:25 ipip
         service=smtp
         scheduler=wlc
         persistent=120
         protocol=tcp
         checktype=negotiate
         netmask=255.255.255.255
```

Anexo B, Archivos de configuración en la capa de servidores reales

- Postfix: main.cf

```
smtpd_banner = $myhostname ESMTP $mail_name (Debian/GNU)
biff = no
append_dot_mydomain = no
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_use_tls=yes
myhostname = clustermail.uis.edu.co
mydomain = clustermail.uis.edu.co
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = /etc/mailname
mydestination = $myhostname, clustermail.uis.edu.co, localhost
relayhost =
mynetworks = 127.0.0.0/8, 200.21.228.172
mailbox_size_limit = 512000000
recipient_delimiter = +
inet_interfaces = all
inet_protocols = all
default_transport = smtp
relay_transport = smtp
readme_directory = /usr/share/doc/postfix
html_directory = /usr/share/doc/postfix/html
home_mailbox = Maildir/
mailbox_command = procmail -a "$EXTENSION"
content_filter=smtp-amavis:[127.0.0.1]:10024
```

Anexo C, Archivos de configuración en la capa de almacenamiento

- DRBD : drbd.conf

#Archivo de configuración para una pareja de OST'S, similar al de los servidores de #metadatos cambiando a la respectiva configuración de red.

```
common {
        syncer { rate 90M; }
    }
  resource r0 {
    protocol C;
    on ds3 {
        device     /dev/drbd0;
        disk       /dev/sda3;
        address    192.168.109.29:7787;
        meta-disk  internal;
        }
    on cds3{
        device    /dev/drbd0;
        disk      /dev/sda3;
        address   192.168.109.28:7787;
        meta-disk internal;
        }
    }
```

- Heartbeat: ha.cf

#Este archivo es similar en cada pareja de servidores de almacenamiento

```
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility local0
keepalive 2
```

deadtime 10
initdead 30
udpport 693
bcast eth0
auto_failback off
node ds1
node cds1
ping 192.168.109.1
respawn hacluster /usr/lib/heartbeat/ipfail

- Heartbeat: haresources

ds1 192.168.109.199 drbddisk::r0 mlustre

- Heartbeat: mlustre

#script para montar lustre luego de levantar su respectiva copia DRBD
mount -t lustre /dev/drbd0 /mnt/lustre/mdt
exit 0

- Heartbeat: auhtkeys

auth 2
2 sha1 clusterha1

Anexo D, Encuesta sobre la funcionalidad el servicio de correo.

1. ¿Cuando intentó acceder al servicio este estuvo disponible? SI___ NO___.
2. ¿Pudo realizar correctamente el envío de mensajes a otros servidores de correo en Internet como Gmail y Hotmail? SI___ NO___.
3. ¿Pudo realizar correctamente la recepción de mensajes desde otros servidores de corroer en Internet como Gmail y Hotmail? SI___ NO___.
4. ¿Notó algún comportamiento anormal durante la utilización del servicio? SI___ NO___ ¿Cuál? Cierres inesperados___ Respuesta errónea del sistema___ Otro_____.
5. ¿Le resultó práctica la interfaz Web del servicio de correo? SI___ NO___.

Anexo E, Descripción librerías para las pruebas sobre los sistemas de archivos

- **Librería ROMIO**

ROMIO es una implementación de funciones para manejo de E/S de alto rendimiento integrado a MPI/IO de la versión 2.0 en adelante. Estas funciones pueden ser utilizadas sobre distintos tipos de sistemas de archivos (NFS, PVFS, Lustre, ext3, etc.). Se caracteriza por la utilización de dos técnicas para el manejo de datos:

- *Data sieving,* técnica para acceder eficientemente a bloques de datos no contiguos cuando las funciones primitivas del archivo utilizado no pueden hacerlo. La dinámica de funcionamiento consiste en acceder a las regiones de datos no contiguas accediendo un bloque de datos que contenga todas las regiones deseadas incluyendo los datos que no se necesitan. Las regiones de interés serán extraídas en el cliente. Esta técnica tiene la ventaja de realizar una única transferencia de datos pero accede más datos de disco que los requeridos y transfiere más datos por la red.
- *Two phase I/O* o collective buffering, es una optimización para operaciones colectivas de E/S. El conjunto de E/S independientes que hacen parte de la operación colectiva, determina que regiones deben ser transferidas (leídas o escritas). Dichas regiones serán distribuidas entre los procesos que finalmente interactuarán con el fichero. Por ejemplo para la escritura, los datos son recolectados primero por el proceso que pide la escritura y luego son tomados por los procesos encargados de realizar efectivamente la escritura a disco.

- **Perf de ROMIO**

Se utilizó el programa perf para evaluar el rendimiento de las operaciones de

lectura y escritura de archivos en paralelo, provistas por ROMIO de MPI I/O.

El programa perf es parte del código fuente de ROMIO. Realiza operaciones de lectura y escritura concurrentes a un mismo archivo. Para cada proceso MPI, perf asigna un arreglo de datos de tamaño fijo, el arreglo es escrito en una región diferente y fija en el archivo compartido utilizando la función MPI File write() y luego es leída de la misma región MPI File read() como se muestra en la ilustración 36 en la cual cada nodo escribió una parte(regiones sombreadas) del archivo. Todos los procesos MPI utilizan la función MPI Barrier() para sincronizar antes de cada operación de E/S. Perf mide 2 tipos de operaciones: escrituras y lectura del archivo, escrituras con sincronismo con la función MPI File sync() y posterior lectura del archivo.

La función MPI File sync() hace que en el momento de su ejecución los datos sean transferidos de memoria al fichero que se esté accediendo. Este programa provee una cota máxima de rendimiento de operaciones de E/S de MPI según la cantidad nodos por corrida y del tipo de sistema de archivos utilizado.

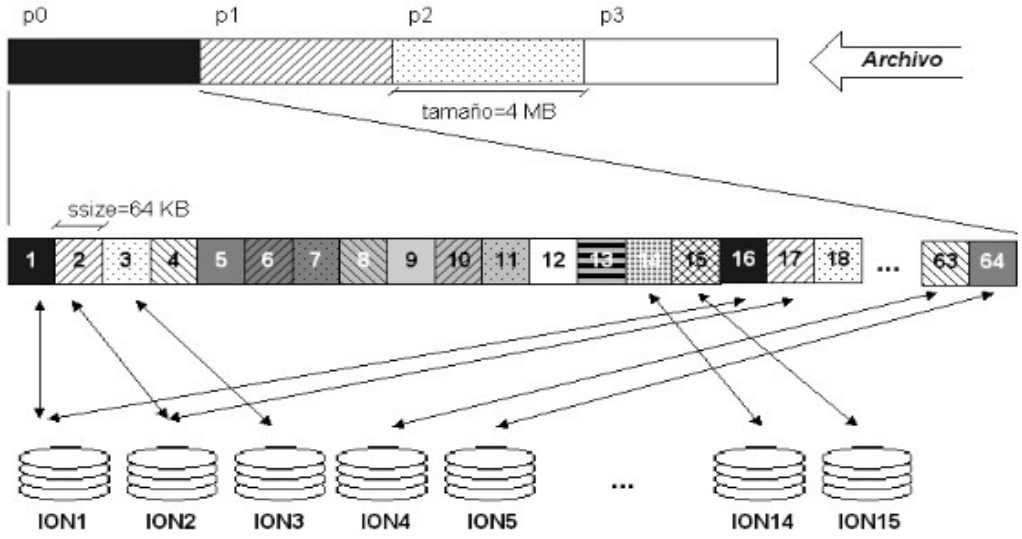


Ilustración 36: Distribución física del archivo generado por perf para la ejecución con 4 nodos de tamaño 4 MB por nodo.

Fuente: Análisis de performance y optimización en Clusters Beowulf Maria Varolina León Carri

- **Mkrandpfile**

El programa mkranpfile[26] es un benchmark escrito en C y MPI IO, realizado por Nikos Drakos en la Computer Based Learning Unit, University of Leeds, UK. Este benchmark escribe archivos separados; un archivo por cada uno de los procesos de MPI que ejecuta y son realizados en paralelo. El tamaño del archivo a crear se define en la variable BLOCK_SIZE.

[26] High Performance Data Management and Processing http://beige.ucs.indiana.edu/I590/node86.html