Universidad de las Ciencias Informáticas

Facultad 6

SOLUCIÓN DE CLÚSTER DE BASES DE DATOS PARA SISTEMAS QUE GESTIONAN GRANDES VOLÚMENES DE DATOS UTILIZANDO POSTGRESQL.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autor: Mario Raúl Rico Rodríguez

Tutores: Ing. Leonel Fuentes Marrero.

Ing. Héctor Miguel Beltrán Lugo.

Co-Tutor: Adrian Misael Peña.

Junio, 2011

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Facultad 6 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Mario Raúl Rico Rodríguez

_____

Firma de Autor

Leonel Fuentes Marrero

_____

Firma de Tutor

Datos de contacto

Ing. Leonel Fuentes Marrero: Ingeniero en Ciencias Informáticas.
Centro de desarrollo: **DATEC**
Dirección: Universidad de las Ciencias Informáticas
E-mail: lfmarrero@uci.cu

Ing. Héctor Miguel Beltrán: Ingeniero en Ciencias Informáticas.
Centro de desarrollo: **DATEC**
Dirección: Universidad de las Ciencias Informáticas
E-mail: hmbeltran@uci.cu

Ing. Adrian Misael Peña: Ingeniero en Ciencias Informáticas.
Centro de desarrollo: **DATEC**
Dirección: Universidad de las Ciencias Informáticas
E-mail: ampena@uci.cu

AGRADECIMIENTOS

A mi familia en especial a mi madre por el apoyo que me brindó.

A mis amigos que me acompañaron durante estos cinco años.

A los tutores de esta investigación, en especial a Adrian por ser guía de la tesis, a Bonne, Marcos, Anthony por las constantes preguntas, en general a los cuatros por preocuparse tanto por este trabajo.

A todos que de una forma u otra hicieron posible que pudiera desarrollar esta investigación.

DEDICATORIA

A mi familia.

A los profes que influyeron en mi formación profesional.

A la UCI, a la Revolución.

“(...) Después de escalar una montaña muy alta, descubrimos que hay muchas otras montañas por escalar (...)”

Nelson Mandela

RESUMEN

Cuba está inmersa en el proceso de informatización de la mayoría de los sectores de la sociedad, por lo que adjunto a esto, han surgido un grupo de aplicaciones y sistemas para prestar servicios a dicho objetivo.

Muchas de estas aplicaciones se implementan en la Universidad de las Ciencias Informáticas, contribuyendo con la misma varias entidades del país. En su gran mayoría la implementación de estos sistemas se realiza utilizando *software* libre, siendo su explotación de forma centralizada.

La meta y principal objetivo es implementar una solución que aumente la capacidad de respuesta y disponibilidad en los servidores que manejan grandes volúmenes de datos, utilizando el *SGBD* PostgreSQL, mediante la implementación de un clúster de alto rendimiento y alta disponibilidad.

La solución propuesta contribuyó a un aumento de la disponibilidad y una disminución de los tiempos de respuestas en un cierto grupo de aplicaciones, disminuyendo así el consumo de recursos ejercido sobre los servidores de bases de datos.

Palabras claves Clúster, capacidad de respuesta, clúster de alto rendimiento, clúster de alta disponibilidad.

TABLA DE CONTENIDOS

INTRODUCCIÓN

Las Tecnologías de la Información y las Comunicaciones (TICs) hoy en día juegan un papel fundamental en el desarrollo del mundo. La industria del software cada vez se hace más poderosa y junto con esto ha alcanzado un nivel mayor de madurez en el desarrollo de aplicaciones informáticas.

Cuba no está exenta de esto, aparejado al desarrollo que se ha alcanzado a nivel mundial se han creado alternativas para salir al paso a esta avalancha. La Universidad de las Ciencias Informáticas (UCI), surgida al calor de la Batalla de Ideas, nace con el propósito de favorecer dicho desarrollo, contribuyendo a la informatización de las instituciones del país.

Muchas de las aplicaciones informáticas que se desarrollan en la UCI manejan una gran cantidad de información la cual va creciendo considerablemente. Esta información es importante preservarla en el tiempo por lo que es almacenada en bases de datos para una posterior consulta o análisis. Un ejemplo de estas aplicaciones son los almacenes de datos.

Debido a la gran cantidad de información a procesar mencionada anteriormente y la arquitectura centralizada que presentan dichas bases de datos lo cual conlleva a un alto nivel de concurrencia provocan un alto consumo de recursos en los servidores donde se encuentran alojadas dichas bases de datos. Esto trae como consecuencia que la carga de trabajo de los mismos sea mayor por lo que los recursos que manejan deben ser suficientes para soportarla. En la mayoría de los casos no se cuenta con el hardware necesario para satisfacer estos requerimientos o en el mejor de los casos se puede mejorar este pero hasta un límite.

La explotación de este tipo de bases de datos utilizando servidores con capacidad de procesamiento inferior a los requerimientos así como el aumento de la información de las mismas repercute directamente en el rendimiento del sistema, provocando retardo en el tiempo de respuesta, así como dichos sistemas pueden dejar de estar disponibles por una sobrecarga. Este retraso en el tiempo de respuesta por sobrecarga del servidor es uno de los problemas principales en los entornos que procesan gran cantidad de información.

Como una posible solución a estos problemas de rendimiento en la UCI se han implementado soluciones de clúster de bases de datos, dichas soluciones centran su rendimiento en el balanceo de

carga lo cual no es factible para manejar grandes volúmenes de datos pues se hace de forma ineficiente ya que la manipulación de dichos datos se hace con el mismo volumen en un mismo servidor manteniendo el mismo consumo de recursos para dicho servidor.

Para dar solución a esta problemática la presente investigación está enfocada a dar una solución factible para sistemas que utilicen un gran volumen de datos utilizando el SGBD PostgreSQL, para lo cual se plantea el siguiente problema a resolver: ¿Cómo aumentar el rendimiento y la disponibilidad de grandes volúmenes de datos sobre PostgreSQL?

El objetivo de la investigación es: Diseñar una solución de clúster que aumente la capacidad de respuesta y disponibilidad de los servidores de base de datos PostgreSQL que son utilizados por sistemas que gestionan grandes volúmenes de datos.

Como objeto de estudio de la presente investigación se define: los servidores de bases de datos PostgreSQL.

El campo de acción del objetivo anterior es: la capacidad de respuesta y la disponibilidad de los servidores PostgreSQL.

Para dar cumplimiento a los objetivos planteados anteriormente se definen los siguientes objetivos específicos:

- Realizar un estudio de las técnicas para la implementación de soluciones de clúster sobre el sistema de gestión de bases de datos PostgreSQL y de las herramientas libres que pueden ser utilizadas.
- Diseñar una propuesta de solución de clúster sobre el sistema de gestión de bases de datos PostgreSQL para los sistemas que gestionan grandes volúmenes de datos.
- Realizar pruebas que permitan validar la propuesta de solución que se diseñe.

Como posibles resultados del presente trabajo se espera obtener el diseño de una propuesta de solución para montar un clúster de base de datos utilizando el sistema de gestión de bases de datos PostgreSQL, que va a aumentar las prestaciones de los sistemas que gestionan grandes volúmenes de datos así como manuales necesarios para el montaje y mantenimiento de la solución que se diseñe.

Estructura del documento:

El presente documento se estructura en: Resumen, Introducción, varios capítulos que constituyen el cuerpo de la tesis, Conclusiones, Recomendaciones, Referencias bibliográficas, Anexos y Glosario de Términos.

Los capítulos son:

CAPITULO 1. Fundamentación teórica: En este capítulo se realiza y se muestra un estudio acerca todo lo relacionado con la tecnología de clúster, así como las herramientas necesarias para su configuración.

CAPITULO 2. Propuesta de diseño de la solución de clúster: se describe el diseño lógico y físico de la solución además del diseño de interconexión entre los nodos. También se tratan los temas referentes a la seguridad y el monitoreo.

CAPITULO 3. Diseño de las pruebas y análisis de los resultados: se describe todo el procedimiento de las pruebas de configuración, las funcionalidades del sistema a probar, así como las pruebas que se le aplican a la propuesta de diseño genérico de la solución.

1 CAPÍTULO FUNDAMENTACIÓN TEÓRICA

1.1 Conceptos fundamentales.

Bases de datos: no son más que un conjunto de datos consistentes y usualmente persistentes, organizados en un modo específico que permite acceder a la información de forma fácil y rápida. (Puentes, 2010)

Sistema gestor de bases de datos: es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos. (CAVSI, 2010)

1.2 PostgreSQL.

PostgreSQL es el Sistema Gestor de Bases de Datos relacional de código abierto más avanzado del mundo, lleva más de 20 años de desarrollo, presenta un ciclo de desarrollo bien definido al estilo de FeeBSD, es distribuido bajo licencia BSD (Berkeley Software Distribution) y es adaptable a las necesidades del cliente.

PostgreSQL dispone de soporte para más de 30 plataformas incluyendo a Linux, UNIX, HP-UX, Solaris, Windows, entre otras. Presenta soporte de todas las características de una base de datos profesional: triggers, vistas, procedimientos almacenados, funciones y tipos de datos definidos por el usuario, búsqueda de textos completos. Cumple totalmente con ACID (Atomicity, Consistency, Isolation, Durability).

Incluye la mayoría de los tipos de datos de SQL92 y SQL99, se pueden escribir funciones en más de 10 tipos de lenguajes (PL/pgSQL, PL/Ruby, PL/R, C/C++, PL/Python, PL/Perl, PL/Java).

Tiene soporte para bases de datos geo-referenciales (PostGis). Presenta soporte para la comunicación encriptada a través de SSL. Soporta la internacionalización de aplicaciones, codificaciones de caracteres multibyte, unicode. Realiza ordenaciones dependiendo de la configuración de idioma local y la diferenciación de mayúsculas y minúsculas y del formato.

PostgreSQL ofrece sofisticadas características tales como control concurrente multiversión (MVCC), point in time recovery (PITR), tablespaces, replicación asíncrona, transacciones anidadas (savepoints), copias de seguridad en caliente/en línea, un sofisticado planificador/optimizador de consultas y write ahead logging para ser tolerante a fallos de hardware, además presenta utilidades para la limpieza de las bases de datos (VACUUM).

Es altamente escalable tanto en la cantidad bruta de datos que puede manejar como en el número de usuarios concurrentes que puede atender. Hay sistemas activos en producción con PostgreSQL que manejan más de 4 terabytes de datos. (Ernesto, 2010)

PostgreSQL presenta algunos límites en cuanto a sus bases de datos, a continuación se muestran algunos: (Ernesto, 2010)

- Máximo de base de datos: ILIMITADO
- Máximo de tamaño de tabla: 32 TB
- Máximo de tamaño de registro: 1.6 TB
- Máximo de tamaño de campo: 1GB
- Máximo de registros por tabla: ILIMITADO
- Máximo de campos por tabla: 250 a 1600 (depende de los tipos usados)
- Máximo de índices por tabla: ILIMITADO

1.3 Introducción a la tecnología de clúster.

El volumen de información que manejan los sistemas informáticos en la actualidad aumentan considerablemente cada día, esto provoca que la manipulación de esta sea cada vez más costosa en cuanto a recursos de hardware y en ocasiones las prestaciones necesarias no se logran con una simple computadora.

Al acceder un número significativo de usuarios de manera concurrente a la aplicación ésta debe ser capaz de atender todas las solicitudes en un tiempo aceptable. Aplicaciones como esta, limitadas por los recursos de hardware tienen un rendimiento bajo, una posible solución sería aumentar la capacidad de cómputo con nuevos componentes de hardware o sustituir la computadora por una con mejores condiciones, esto trae consigo un costo adicional al del software así como al del mantenimiento del hardware.

Existe otra forma de aumentar el desempeño de los servidores, la implementación de un clúster de servidores. Esta alternativa es tan potente como una supercomputadora además de ser más rentable económicamente y brindar las mismas funcionalidades con mayor flexibilidad.

1.4 Clúster.

Un clúster: es un conjunto de computadoras, a menudo con semejantes componentes de hardware, que se interconectan entre sí a través de un sistema de red de alta velocidad y son capaces de elevar la eficiencia para realizar determinadas tareas que individualmente no podrían realizar debido a la creciente necesidad de potencia computacional que demandan algunas aplicaciones. (Almaguer, 2008)

Ventajas de esta tecnología.

El uso de clústeres ofrece un sinnúmero de ventajas que garantizan el funcionamiento óptimo del sistema. Además de mejoras en el rendimiento, reducción del tiempo de cálculo y los recursos necesarios, incrementa la flexibilidad con software y hardware. Proporciona gran capacidad computacional tales como el aumento de la velocidad de procesamiento, incrementa el número de transacciones o tiempo de respuesta y el aumenta la confiabilidad de los sistemas.

Los clústeres no solo garantizan el funcionamiento ininterrumpido de algunas aplicaciones ejecutando un mayor número de tareas en el menor tiempo posible, sino que comparten la carga de tráfico y el

procesamiento entre sus nodos. Esto proporciona un entorno confiable que puede ser escalado de una manera simple y rápida a medida que las demandas de la carga de trabajo crezcan.

Otras de las ventajas son los costos de implantación mejorando la relación costo/beneficio, para lograr los mismos resultados de procesamiento se necesitaría una supercomputadora la cual es sumamente cara en comparación con el costo del clúster, que además este se implementa con simples ordenadores, es por esto que un clúster es considerado un supercomputador.

1.5 Soluciones de clúster.

En el mundo existe una tendencia al aumento del uso de este tipo de tecnología para servidores de bases de datos. Un ejemplo de esto es el clúster de aplicaciones de la compañía Google el cual cuenta con más de 60 000 servidores repartidos en diferentes localidades del mundo (Norteamérica, Asia y Europa). (Barroso, 2010)

Cada uno de los servidores de datos de Google tiene instalado un sistema operativo Linux y sobre ellos se realizan técnicas de balanceo de carga y replicación. El uso de un sistema distribuido de almacenamiento de datos no centralizado garantiza un menor costo en la implementación del clúster y reduce las posibilidades de ocurrencia de fallos, por otra parte aumenta la escalabilidad y el rendimiento del sistema lo cual hace posible que puedan ser atendidas hasta 40 millones de búsquedas por día. (Barroso, 2010)

Otra solución de clúster en el mundo es la que ofrece Oracle Real Application Clusters (Oracle RAC), con Oracle Database 11*g* Enterprise Edition, permite ejecutar una sola base de datos en un grupo de servidores y proporciona una tolerancia a fallos, un rendimiento y una capacidad de ampliación inigualables, sin necesidad de cambios de aplicaciones. (Oracle, 2010)

1.6 Soluciones de clúster en Cuba.

En Cuba no hay un grado de madurez alto en el uso de estas tecnologías para bases de datos, muy pocas empresas como ETECSA y la ADUANA las usan, esta última sobre software propietario.

Otro ejemplo del uso de la tecnología de clúster pero esta vez no de bases de datos es el Centro de Ingeniería Genética y Biotecnología que cuenta con un clúster conformado por 128 procesadores,

sistema dedicado a numerosas tareas de procesamiento dentro de la investigación científica de la institución. (Almaguer, 2008)

1.7 Clasificación de los clústeres.

Un clúster se puede clasificar en cuanto a la diversidad de software y hardware en:

- Homogéneo: La configuración de los nodos en cuanto a hardware y software son las mismas.
- Semi homogéneo: La configuración de los nodos difiere en cuanto a hardware o software.
- Heterogéneo: La configuración de los nodos difiere en cuanto a hardware y software.

Dependiendo de las características y el objetivo para el cual fue concebido, un clúster puede clasificarse de las siguientes formas:

Clúster de Alto Rendimiento (HPC): se ejecutan tareas que requieren de gran capacidad computacional, grandes cantidades de memoria, o ambos a la vez. Este tipo de clúster divide las tareas en tareas más pequeñas y las reparte entre los nodos que lo conforman para que sean calculadas entre ellos y así agilizar el procesamiento de los datos. Estos sistemas se implementan en un ambiente de programación paralela utilizando algoritmos que hacen uso de recursos compartidos tales como CPU, memoria, datos y servicios. (Almaguer, 2008)

Clúster de Alta Disponibilidad (HAC): tiene el objetivo de proporcionar máxima disponibilidad y confiabilidad de los servicios que ofrece, de tal manera que estos se brinden ininterrumpidamente. Se consigue alta disponibilidad haciendo redundantes los sistemas y de esta forma cuando se produce la caída del nodo que brinda el servicio un clon del mismo puede tomar el control y comenzar a servir el servicio nuevamente. La confiabilidad se consigue mediante un software que detecta fallos y permite la recuperación frente a los mismos. Un clúster de alta disponibilidad evita que el sistema tenga un único punto vulnerable a la ocurrencia de fallos. (Almaguer, 2008)

Clúster de Balanceo de Carga (LBC): se encarga de colocar en paralelo varios servidores (servidores reales) capaces de brindar el mismo servicio y de alguna forma repartir el trabajo entre ellos, tal que los clientes vean servidas sus solicitudes en tiempos menores y aceptables. Los clientes deben ver al conjunto de servidores como si fuera uno solo (servidor virtual). Un clúster de balanceo de carga tiene peculiaridades de clúster de alta disponibilidad y alto rendimiento. (Bourke, 2010)

Teniendo en cuenta las clasificaciones anteriores el clúster que se propone en esta investigación es: homogéneo y presenta una combinación de alto rendimiento con alta disponibilidad para obtener una mayor capacidad de procesamiento y tolerancia a fallos.

A partir de la clasificación del clúster propuesto en la investigación este debe realizar las siguientes funcionalidades:

- Particionado de datos.
- Replicación.
- Alta disponibilidad.

Como tal PostgreSQL no provee una solución completa para clúster, muchas de las funcionalidades necesarias se logran utilizando una serie de herramientas externas al gestor que en su conjunto conforman la solución, a continuación se abordarán las herramientas que se utilizan para esto. (Almaguer, 2008)

1.8 Herramientas para configurar el clúster propuesto.

Alta disponibilidad.

En la actualidad las organizaciones dependen cada vez más de sus sistemas de información, y como es obvio se desea que estos sean seguros y permanezcan disponibles el mayor tiempo posible. (Clavijo, 2010)

Teniendo en cuenta lo anteriormente expuesto, la alta disponibilidad no es más que la capacidad que tiene un sistema de estar prestando servicio las 24 horas del día durante los 7 días de la semana.

Heartbeat: esta herramienta es una de las principales del proyecto Linux-HA, el objetivo principal de este proyecto es proporcionar soluciones de alta disponibilidad para Linux. Es el paquete por excelencia de alta disponibilidad para Linux que soporta desde 2 a “n” nodos, permite crear grupos de recursos y cambiarlos fácilmente. Esta herramienta permite crear un clúster de alta disponibilidad asegurando que todos los servicios se ejecuten en todo momento realizando un intercambio de paquetes a través de los enlaces de red o series, además estos clústeres son de control

descentralizado. El principio de funcionamiento de esta herramienta consiste en el envío de latidos (ping) por la red al servidor principal verificando que esté activo, estos envíos constantes de ping requieren de una respuesta por parte del servidor, cuando transcurre un tiempo y el servidor principal no responde Heartbeat determina que el mismo está fuera de servicio o está inactivo, automáticamente este activa el servidor secundario de forma que asuma todas las peticiones del cliente sin afectar la aplicación. (Suárez, 2010)

Pacemaker: es un manejador de recursos de código abierto, el cual se integra con Heartbeat. Es el encargado de actuar en caso de haber un fallo en cualquier servicio del clúster. Además se pueden crear recursos y grupos de recursos los cuales se pueden iniciar, detener, monitorear. (Clusterlabs, 2010)

Ldirectord (Linux Directord Daemon): es un demonio para monitorear los servicios ofrecidos en los servidores reales en un clúster LVS. Su funcionamiento consiste en hacer un chequeo constante y verificar la disponibilidad de los servicios que prestan los servidores reales del clúster. Cuando Ldirectord detecta la caída de alguno de los servicios monitoreados en uno de los servidores reales este hace una reconfiguración automática de la tabla de rutas, utilizando el software Ipvsadm, para que el balanceador deje de enviar paquetes hacia este servidor.

Como herramienta para garantizar la alta disponibilidad se selecciona Heartbeat ya que permite la integración con Pgpool-II y las herramientas anteriores provienen de la solución de clúster que ofrece Red Hat (LVS) el cual es propietario.

Replicación.

La replicación no es más que la técnica que permite copiar y distribuir idénticamente las tablas de una base de datos en múltiples bases de datos ubicadas en diferentes nodos de la red. La replicación asegura que los datos correctos estén siempre disponibles en el momento y en el lugar necesario. (Freedman, 2010)

PostgreSQL: en su versión más reciente 9.0 incorpora una serie de nuevas funcionalidades entre las que se encuentra Streaming Replication. Esta nueva funcionalidad da la posibilidad de tener un sistema de replicación asincrónica maestro-esclavos. Su funcionamiento consiste en transferir asincrónicamente registros WAL sobre la marcha (record-based log shipping) entre un servidor

maestro y uno o varios esclavos. Una de las desventajas que presenta este tipo de replicación es que replica el clúster de datos completo, o sea todas las bases de datos del servidor. (PostgreSQL Development Group, 2010)

Pgpool-II: es un intermediario entre servidores de bases de datos PostgreSQL y sus clientes totalmente transparente. Trabaja sobre los protocolos frontend y backend de PostgreSQL pasando las conexiones entre ellos, por lo cual una aplicación o cliente (frontend) cree que Pgpool-II es el verdadero servidor de bases de datos y a su vez el servidor (backend) de bases de datos cree que Pgpool-II es el cliente. Esta herramienta funciona en la mayoría de las arquitecturas UNIX, Linux y en Windows no está soportado. Soporta cualquier versión de PostgreSQL a partir de la 6.4. Pgpool-II proporciona un conjunto de funcionalidades como son: replicación, pool de conexiones, balanceo de carga, paralelización de consultas y limita el excedente de conexiones. (Pgpool Home Page, 2010)
En el modo Replicación Pgpool-II actúa como agente de replicación, para lo que envía las consultas de modificación de datos a todos los nodos del clúster y las de selección las distribuye entre ellos. (Sabater, 2010)

Como herramienta para realizar la replicación se selecciona Pgpool-II ya que la misma permite replicar tablas específicas a diferencia de la otra.

Particionado de Datos.

Particionar se refiere a la división de lo que es lógicamente una gran tabla en pequeñas tablas físicas.

El particionado puede ofrecer varias ventajas:

- ✓ El rendimiento de las consultas se puede mejorar sustancialmente en ciertas situaciones, sobre todo cuando la mayoría de las filas más visitadas de la tabla están en una sola partición o un pequeño número de particiones.

- ✓ Cuando las consultas o las actualizaciones acceden a un gran porcentaje de una sola partición el rendimiento puede ser mejorado mediante el aprovechamiento de la exploración secuencial de esa partición en lugar de utilizar un índice y lecturas de acceso aleatorio esparcidos por toda la tabla.

- ✓ Cargar y eliminar masivamente se pueden lograr mediante la adición o eliminación de particiones, siempre que dicho requisito está previsto en el diseño de particiones. ALTER TABLE es mucho más rápido que una operación masiva. También todo esto evita el vacío sobrecargado causado por un DELETE masivo.

- ✓ Los datos utilizados rara vez se pueden migrar a los medios de almacenamiento más baratos y más lentos.

Los beneficios normalmente valdrán la pena solo cuando una tabla esté particionada, de otro modo será muy grande. El punto exacto en el que una tabla se beneficiará de la partición depende de la aplicación, aunque una regla general es que el tamaño de la tabla debe exceder la memoria física del el servidor de base de datos (PostgreSQL Development Group, 2010).

PostgreSQL: tiene la facilidad de implementar particionado mediante herencia de tablas, dicha herencia tiene una tabla padre la cual no contiene datos, los cuales son almacenados en las tablas hijas cuando son manipulados. (PostgreSQL Development Group, 2010)

PL/Proxy: es un lenguaje compacto para llamadas remotas entre bases de datos PostgreSQL. La sintaxis es similar a PL / pgSQL y el usuario puede crear funciones de proxy que tiene la misma forma que las funciones de control remoto para ser llamadas. (PL/Proxy Project Home Page, 2010)

Pgpool-II: en el modo paralelización de consultas permite dividir los datos entre múltiples servidores de modo que una consulta puede ser ejecutada en todos los servidores al mismo tiempo para reducir el tiempo de ejecución global. Esto lo realiza mediante funciones de distribución definidas por el usuario tanto para la inserción de los datos como para la selección. (Pgpool Home Page, 2010)

Como herramienta de particionado de datos se selecciona Pgpool-II por la facilidad de particionar tablas en diferentes nodos a demás de trabajar a nivel de consultas a diferencia de PL/Proxy que lo hace por funciones.

1.9 Pruebas necesarias para la solución propuesta.

Con el objetivo de garantizar el correcto funcionamiento de cualquier sistema es necesario realizar un proceso de pruebas donde se demuestre que dicho sistema está listo para su despliegue así como si su funcionamiento es óptimo.

Pruebas de **carga:** tienen como objetivo analizar el comportamiento de una aplicación al ser sometida a los diferentes niveles de carga previstos para las operaciones de producción. (Meier, 2010) Permiten definir los límites operacionales de una configuración para los diferentes niveles de carga.

Pruebas de **configuración:** estas pruebas se realizan para garantizar que la aplicación funcione correctamente sobre diferentes configuraciones de hardware y/o software. Durante su realización tiende a ocurrir un ciclo de perfeccionamiento en el diseño de la solución. (Rational Unified Process, 2010)

Pruebas de **benchmark:** este tipo de prueba permite comparar el rendimiento de un nuevo o desconocido objeto de pruebas respecto al de un sistema o referencia conocida. (Rational Unified Process, 2010) Es muy utilizado para definir entre diferentes arquitecturas de despliegues o configuraciones específicas cual es la mejor para dar solución a un determinado problema.

Las pruebas que se le van a realizar a la solución propuesta son las de carga y configuración a fin de validar el correcto funcionamiento de la aplicación así como determinar el nivel de carga que soporta.

Métricas que se utilizarán para las pruebas.

Para evaluar el rendimiento de una aplicación, es necesario hacer mediciones del comportamiento de algunas de sus características. Con estas mediciones se pueden obtener datos de todo el clúster en general, así como de cada uno de sus servidores por separados. Las más comunes son:

- Número de peticiones respondidas por segundo.
- Número de peticiones completadas.
- Porciento de peticiones fallidas.
- Tiempo de respuesta promedio del sistema.

Cuando se trata de analizar el comportamiento de los servidores que forman parte del sistema las mediciones se enfocan en el consumo de los recursos. Las mediciones que normalmente se realizan en los servidores son:

- Porciento de uso del CPU.
- Porciento de consumo de memoria RAM.
- Porciento de uso de la red.

Herramientas para generar carga.

Pgbench: prueba práctica que se incluye dentro de PostgreSQL, prueba de rendimiento muy simple con la que se puede comprobar el subsistema y/o la velocidad a la que se procesan las conexiones. Útil para demostrar problemas importantes tanto en el hardware como en el sistema operativo. Por defecto las pruebas pgbench implican cinco SELECT, UPDATE, y comandos INSERT por transacción. (PostgreSQL Development Group, 2010)

JMeter: es una aplicación de escritorio desarrollada sobre Java. Originalmente fue diseñada para realizar pruebas de carga y medir el rendimiento de aplicaciones web. Su uso se ha extendido a otros tipos de pruebas. Actualmente el JMeter se puede utilizar para probar el rendimiento en recursos estáticos y dinámicos. Con esta herramienta se puede generar una pesada carga a servidores web, ftp y de bases de datos, para analizar su comportamiento bajo diferentes niveles de carga. (Apache Software Foundation, 2010)

Tsung: herramienta que simula un elevado número de usuarios, así como, sesiones diferentes en una base de datos, está desarrollado en Erlang que es un lenguaje diseñado para desarrollar sistemas altamente concurrentes y tolerantes a fallos, robusta y fiable, herramienta de código abierto.

1.10 Herramientas para el monitoreo.

El monitoreo tanto de los servidores como del clúster en su conjunto es de gran importancia en el funcionamiento del mismo, ya que es fundamental administrar adecuadamente los recursos para así detectar fallos a nivel de software o de hardware, además de constatar su correcto funcionamiento y en caso de algún fallo tomar medidas preventivas.

Nagios: es un sistema de código abierto muy configurable y popular que se utiliza para monitorizar las computadoras y servicios de una red. Cuenta con un sistema de avisos para alertar a sus contactos cuando ocurre algún problema en los servicios o computadoras que monitorea, igualmente alerta cuando estos problemas son resueltos. Para enviar los avisos de fallo en tiempo real Nagios hace uso de varios medios como por ejemplo e-mail, jabber, SMS, fax y otros programas externos. (Galstad, 2010)

Sysstat: el paquete Sysstat contiene utilidades para monitorizar el rendimiento del sistema y la actividad del mismo. Sysstat contiene la utilidad sar, común en muchos Unix comerciales y herramientas que se pueden programar vía cron para recoger datos de rendimiento, actividad y mantener un historial. (BLSF Equipo de desarrollo, 2010)

1.11 Otras herramientas

La seguridad es un aspecto importante a tener en cuenta en cualquier sistema informático y más cuando se manejan grandes volúmenes de datos, a continuación se abordará una herramienta muy importante en la seguridad de cualquier sistema que esté conectado a la red.

Iptables: el kernel (núcleo) del sistema operativo GNU/Linux incorpora el manejo de paquetes TCP/IP. Iptables es una herramienta que forma parte del kernel de Linux a partir de su versión 2.4 la cual permite establecer un grupo de reglas de filtrado para indicarle al kernel cómo debe manejar los paquetes TCP/IP.

El manejo de los paquetes se realiza identificando diferentes partes que conforman sus cabeceras TCP (Transmission Control Protocol), e IP (Internet Protocol). De esta manera se pueden confeccionar reglas de filtraje teniendo en cuenta algunas características que los diferencian entre sí tales como la dirección IP origen y destino de los paquetes, el puerto, tipo de servicio, entre otras. Las acciones que el kernel puede realizar sobre los paquetes son de aceptación, negación, rechazo, además puede establecer marcas sobre ellos y modificarlos. (Andreasson, 2010)

La herramienta de monitoreo seleccionada es Nagios por su nivel de adaptabilidad a diferentes entornos y por su facilidad de monitorear más de un servidor de manera simultanea.

1.12 Conclusiones del capítulo.

Durante el desarrollo del capítulo se ha hecho un estudio de los conceptos fundamentales de la tecnología de clúster, las diferentes clasificaciones que existen de los mismos, así como las diferentes herramientas que se utilizan para la configuración de estos.

Para la implementación del clúster propuesto en la investigación se han seleccionado las siguientes herramientas: Pgpool-II como software de replicación y particionado de datos, Heartbeat como software de alta disponibilidad, Nagios como herramienta de monitoreo, JMeter como herramienta para generar carga a la hora de realizar pruebas e Iptables como herramienta de seguridad, en este caso un cortafuegos de red.

2
CAPÍTULO
PROPUESTA DE DISEÑO DE LA SOLUCIÓN DE CLÚSTER

2.1. Diseño de la solución.

Se confeccionó una propuesta de solución para la puesta en marcha de un clúster de servidores de bases de datos. Los nodos utilizados para conformar el clúster pueden ser computadoras personales con características de hardware variables. La ubicación física del mismo debe realizarse en una instalación con buena climatización y acceso restringido al personal.

A continuación se describen los tipos diferentes de diseño:

2.2. Diseño lógico.

Para la implementación de la solución se propone que se utilice como sistema operativo Debian GNU/Linux en su versión 6 (Squeeze), se realizará una instalación mínima del sistema lo cual significa que tendrá solamente las funcionalidades básicas. Se utilizará la versión 9.0.4 de PostgreSQL como SGBD, además como software de particionado y replicación de datos se utilizará Pgpool-II-3.0.3 y para garantizar la alta disponibilidad se empleará Heartbeat-3.

El clúster tendrá la siguiente arquitectura:

- Un nodo Pgpool-II primario.
- Un nodo Pgpool-II segundario de respaldo.
- Dos nodos de datos PostgreSQL, como mínimo dos.

En un estado normal del clúster todas las sentencias SQL que se envíen desde el cliente hacia el clúster pasan por el nodo Pgpool-II en el cual se analizan y determina a cuál nodo PostgreSQL las reenvía en dependencia de las funciones de distribución que se utilicen.

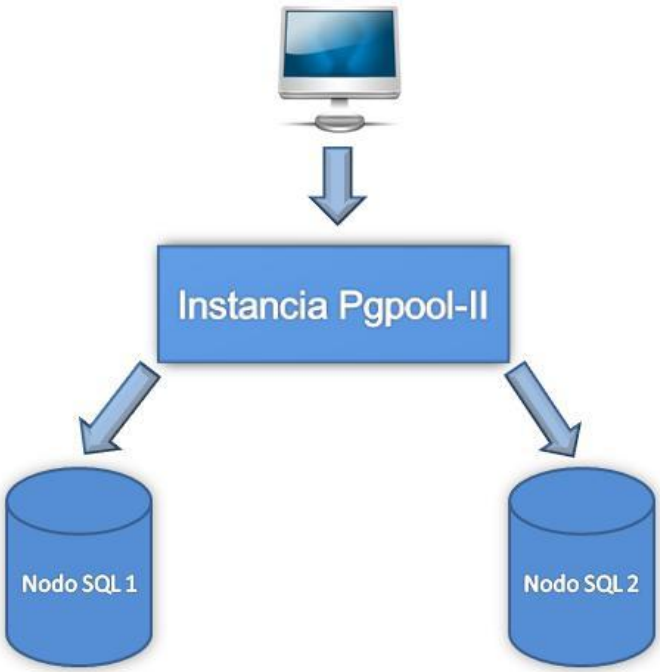

Figura 1. Arquitectura del clúster.

En la arquitectura presentada si el nodo Pgpool-II primario queda fuera de servicio le sucediera lo mismo al clúster en general, por lo cual se convierte en un punto único de fallas el nodo primario.

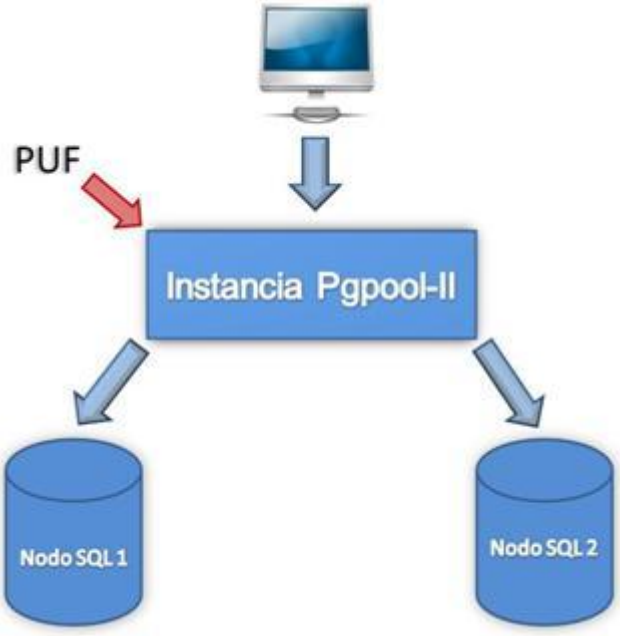

Figura 2: Punto único de fallas.

Para evitar esto se tiene un nodo de respaldo configurado con la alta disponibilidad de Pgpool-II, en caso de caída del nodo primario el de respaldo tomaría el control, siendo este proceso transparente al cliente.

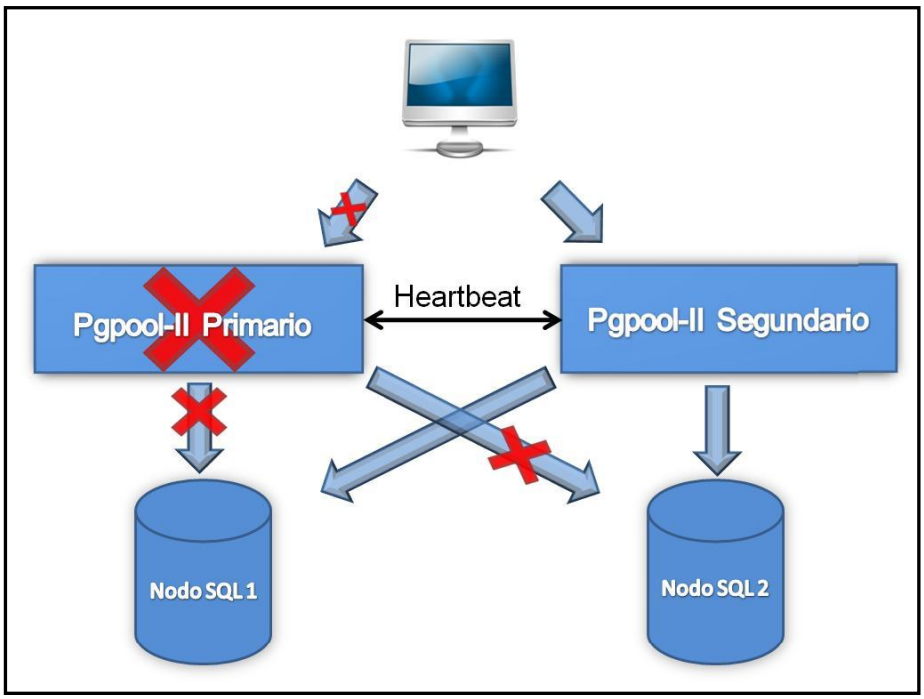

Figura 3: Alta disponibilidad de Pgpool-II.

Cuando el nodo caído se recupere tomaría el control y el de respaldo volvería a ser reserva volviendo todo a la normalidad.

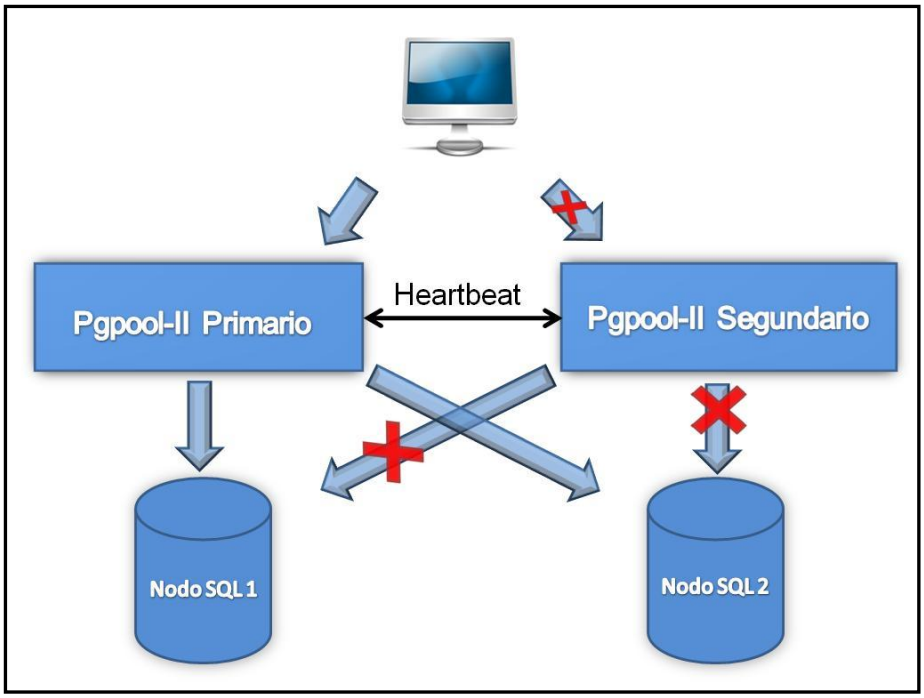

Figura 4: Recuperación de Pgpool-II.

2.3. Diseño físico.

La arquitectura propuesta para el clúster está conformada como mínimo por dos nodos, encargados de fluctuar las peticiones entre los servidores reales, instalándose en uno de ellos la herramienta Pgpool-II-3.0.3, encargada de gestionar la ejecución de las solicitudes provenientes de los clientes.

El sistema de gestión de bases de datos empleado es PostgreSQL-9.0.4, el cual va a recibir las peticiones provenientes de la instancia pgpool-II que en ese momento se encuentre activa. Para ello PostgreSQL tiene que estar configurado para que acepte dichas peticiones.

Un diseño físico básico de la solución en caso que el nodo Pgpool-II esté trabajando de forma normal sería:

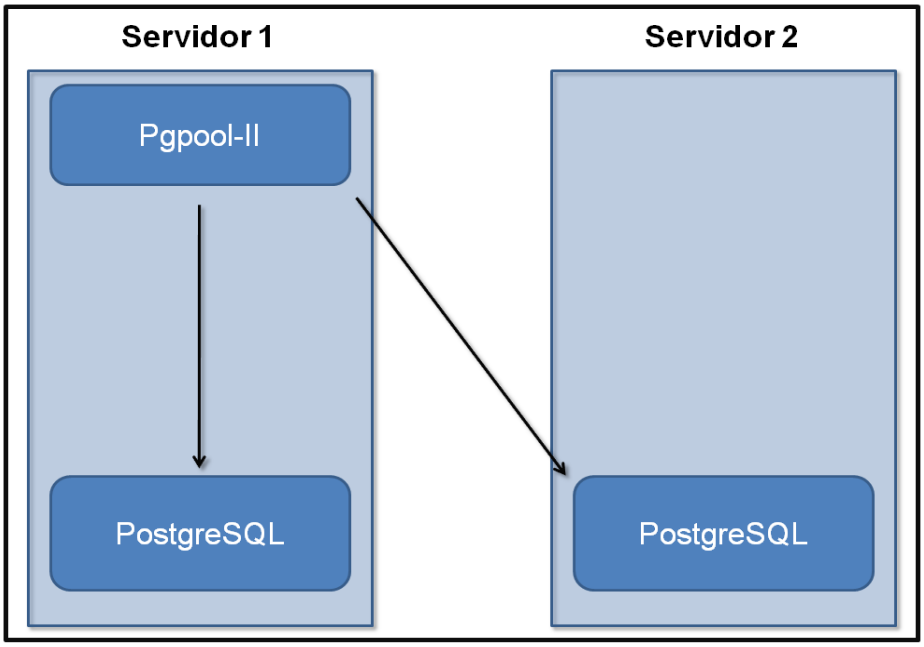

Figura 5: Diseño físico.

La afectación que presenta este diseño es que la instancia del nodo SQL caiga, esto no afectaría en nada la solución pues existe un nodo de reserva que asumiría el control de los datos propios del nodo caído, para que esto sea posible el nodo de reserva tiene que poder acceder a el clúster de datos del nodo caído, por lo que los nodos PostgreSQL tienen que tener su almacenamiento en una SAN (Storage Area Network) que no es más que un conjunto de discos duros interconectados entre ellos por diferentes tecnologías para garantizar la disponibilidad de los datos.

Este diseño presenta un único punto de fallas, ya que el nodo Pgpool-II puede dejar de prestar servicio e imposibilitar la entrada de sentencias SQL por parte del cliente hacia los nodos de datos. Ante este problema gracias a la implementación de alta disponibilidad con Pgpool-II, el nodo de respaldo toma las acciones del servidor principal y continúa prestando servicios totalmente transparente al cliente.

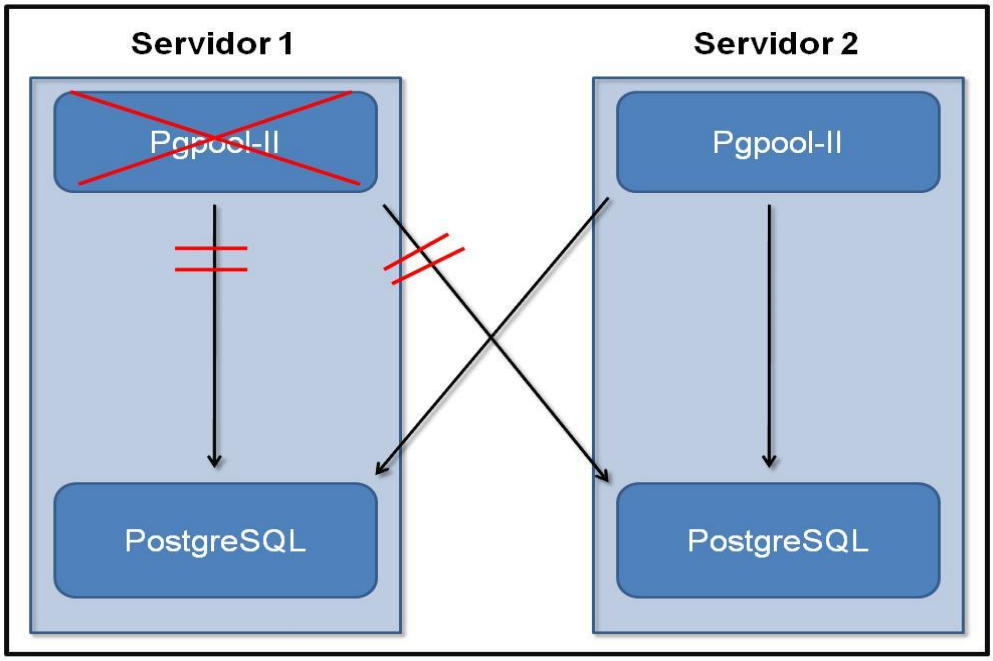

Figura 6: Alta disponibilidad de Pgpool-II.

Pgpool-II brinda la posibilidad de seguir dando servicio tras la caída de N-1 servidores de PostgreSQL en el clúster, esto se logra mediante la integración de Pgpool-II con Heartbeat, dicha herramienta permite detectar la caída completa de un nodo, por lo que entra en la conformación del clúster una nueva dirección IP que sería la dirección de servicio, la cual ambos nodos van a compartir y Heartbeat se encargará de gestionar.

La dirección IP de servicio es gestionada por el sistema de Heartbeat, es movida por el clúster donde los servicios correspondientes se estén ejecutando. Estas direcciones de servicio son direcciones a través de las cuales las aplicaciones y usuarios de los servicios acceden a estos. Es de suma importancia que la dirección IP de servicio no sea gestionada por el sistema operativo, solo que sea gestionada por Heartbeat.

Si se le da una dirección administrativa a Heartbeat para que la gestione, esto causará problemas pues confundirá al sistema. Acarreando como consecuencia que el sistema operativo y Heartbeat se contradigan por el control de la misma.

De esta forma quedaría el diseño general del clúster propuesto:

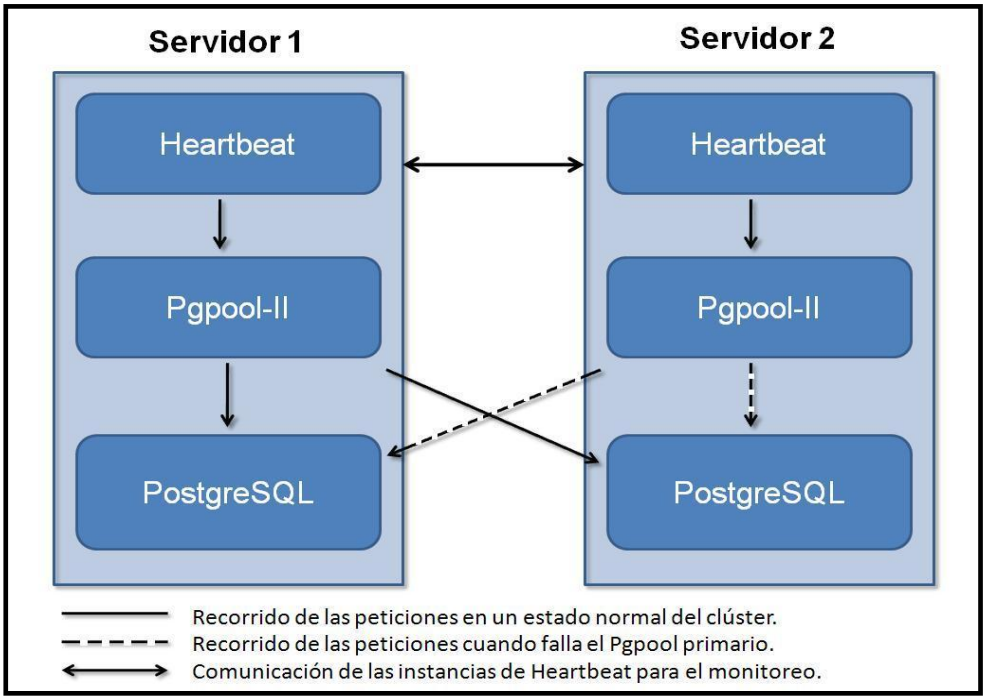

Figura 7: Diseño general del clúster.

2.4. Diseño de interconexión.

El diseño propuesto contempla dos nodos Pgpool-II, de los cuales el principal va a estar activo y por consiguiente recepcionando las sentencias SQL proveniente de los clientes y ejecutándolas en los nodos de datos. Ambos nodos Pgpool-II actúan como intermediarios entre los clientes y los nodos de datos a los cuales serán enviadas las consultas en dependencia del plan de ejecución creado por Pgpool-II.

En caso de caer el nodo principal Pgpool automáticamente tomaría el control el nodo de respaldo. Para lograr esto se utiliza el software Heartbeat el cual debe estar interconectado entre todos los nodos del clúster para así poder detectar la posible caída de alguno de estos.

En el diseño de interconexión entre ambos nodos está propuesto para que se configuren dos tarjetas de red las cuales se describen a continuación:

- Una tarjeta de red: para los servidores por la cual estos van a estar aceptando las peticiones de los clientes/aplicaciones.

- Una tarjeta de red: para que Heartbeat monitoree los nodos.

Debe existir una relación de confianza entre el nodo principal Pgpool-II y los nodos de datos para que este pueda mandar a reiniciarlos cuando los recursos sean migrados a él, para lograrlo es necesario en cada servidor tener instalado el servicio ssh.

Como se mencionó anteriormente los nodos de datos tienen que tener sus datos en una SAN a la cual van a estar interconectados mediante red. De esta forma en caso que el nodo de reserva necesite acceder a los datos de cualquier nodo activo tras su caída lo pueda hacer sin dificultad alguna.

El diseño de interconexión de los nodos de datos a la SAN quedaría de la siguiente forma:

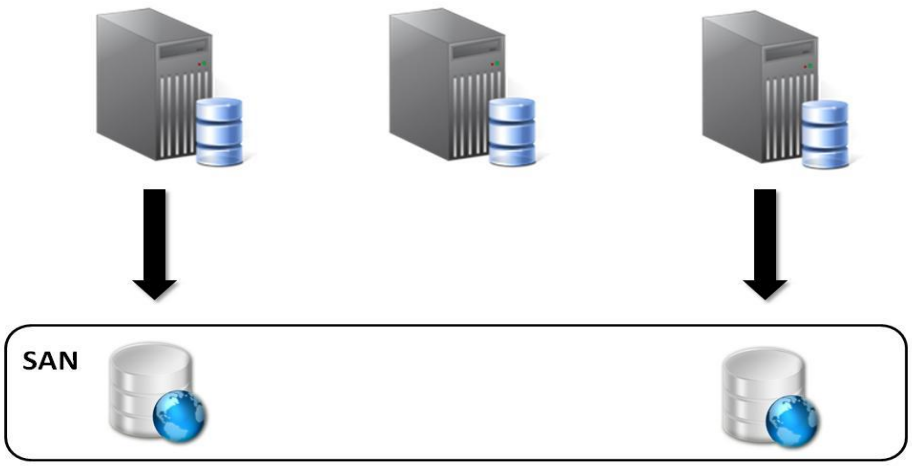

Figura 8: Diseño de interconexión a la SAN.

2.5. Diseño de seguridad.

Uno de los aspectos fundamentales en las soluciones informáticas es el tema de la seguridad hoy en día se producen ataques masivos a dichas aplicaciones y en mayor medida si estas están interconectadas a la red.

Como medida de seguridad se puede optar por la instalación y uso de un cortafuegos para filtrar todas las conexiones tanto entrantes como salientes a dichos servidores, pudiendo aceptar o denegar en dependencia del nivel de seguridad.

Para lograr una efectividad en la configuración de dicho cortafuegos es necesario tener conocimientos básicos de protocolos y servicios de red así como el funcionamiento de las aplicaciones que se quieren proteger.

Existen dos tipos de cortafuegos los cuales son:

Cortafuegos de hardware: se trata de un tipo de cortafuegos instalado en un periférico de aspecto parecido a un router. Es una buena solución cuando lo que se quiere proteger es una red, ya que permite hacer toda la configuración de cortafuego en un solo punto al que se conectan los ordenadores.

Cortafuegos de software: es el más común y utilizado, se trata de un software que se instala en el ordenador, por lo que sólo va a proteger al ordenador en el que está instalado.

En la solución propuesta se va a utilizar un cortafuegos de software ya que el de hardware agrega un costo adicional a la solución.

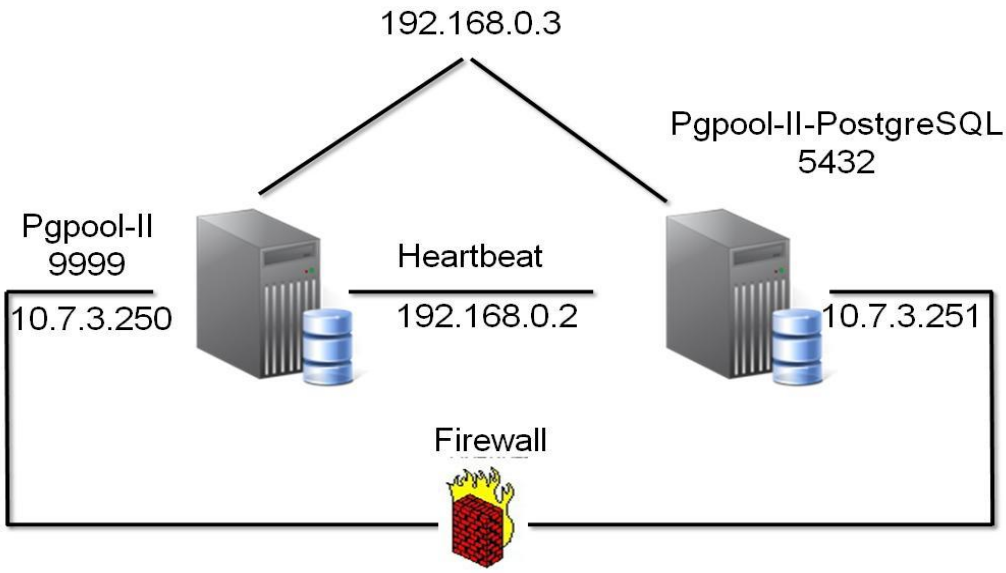

Figura 9: Diseño de seguridad con un cortafuego de hardware.

Como propuesta de solución al tema de los cortafuegos se va a instalar uno de software en cada servidor para filtrar todo el tráfico entrante y saliente del mismo.

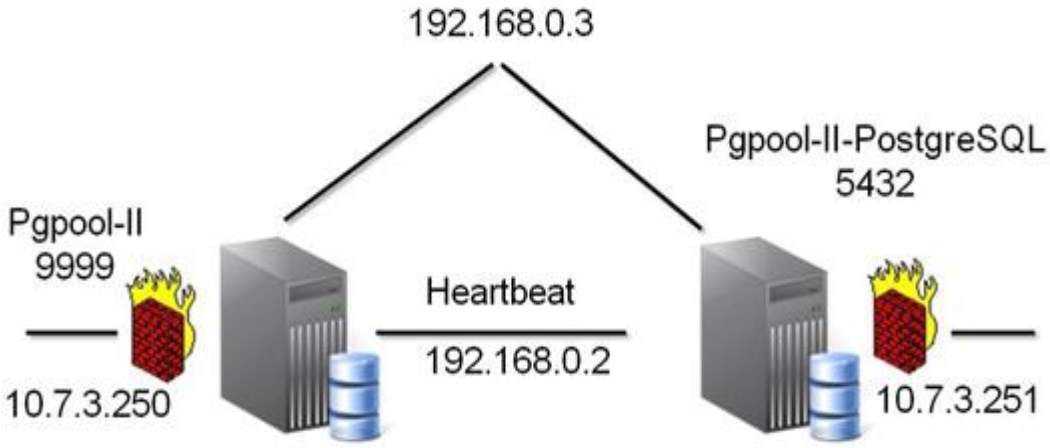

Figura 10: Diseño de seguridad.

Como bien se muestra en la figura anterior el cortafuegos filtrará todo el tráfico hacia los nodos. Para ello Iptables utiliza tres políticas principales: aceptar, denegar o rechazar. En la solución se va utilizará

la combinación de las opciones de denegar todo y aceptar solo las conexiones necesarias a los servicios que brinda el clúster desde las direcciones permitidas. En los nodos se aceptará solamente conexiones a PostgreSQL desde Pgpool-II, a Pgpool-II solamente se aceptarán conexiones desde las direcciones donde se encuentren alojadas las aplicaciones que interactúan con el clúster.

2.6. Diseño de monitoreo.

El monitoreo de la solución se va a dividir en dos partes:

Monitoreo al servidor: en el cual se van a tener en cuenta indicadores relacionados con el consumo de memoria RAM y virtual, actividad del microprocesador, el uso de espacio en disco y la transferencia de red.

Monitoreo a PostgreSQL: en el cual se van a tener en cuenta indicadores relacionados con la cantidad de conexiones activas a la base de datos, cantidad de bloqueos a las bases de datos, cantidad de consultas activas concurrente y tiempo de demora de las consultas que se ejecutan.

La herramienta que se va a utilizar para realizar dicho monitoreo es Nagios, por la gran cantidad de facilidades que presenta. Permite monitorizar más de un servidor a la vez además de la arquitectura cliente-servidor que presenta, la cual permite desde un nodo monitorizar otros sin necesidad de instalarlo en los demás.

Dicha herramienta se va a instalar en una computadora externa al clúster para así desde la misma monitorizar los nodos del clúster mediante el plugins (NRPE, Nagios Remote Plugin Excecutor) que presenta la misma.

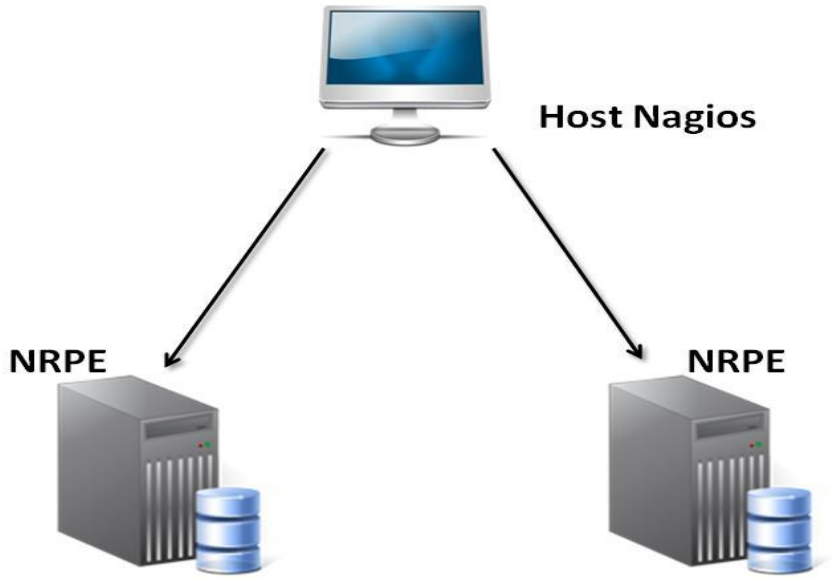

Figura 11: Diseño de monitoreo.

2.7. Implementación de la propuesta de solución.

Para la implementación del clúster de alta disponibilidad y alto rendimiento de bases de datos que se propone en la solución se utilizó como software de particionado de datos Pgpool-II en su versión 3.0.3, el cual recibe todas las peticiones de los clientes y las manda a ejecutar en los nodos PostgreSQL. En dicha solución se busca un aumento del rendimiento a través del particionado de datos por lo que Pgpool juega un papel fundamental, ya que es el encargado de particionar la/las tablas en los diferentes nodos que conforman el clúster a través de la/las funciones de distribución definida por el usuario, además de replicar las otras que no sean particionados para mantener la integridad referencial en tablas relacionadas.

Como SGBD se utilizó PostgreSQL en su versión 9.0.4 que es el encargado de recibir las peticiones enviadas desde Pgpool-II, procesarlas y devolverle el resultado.

La arquitectura final de la solución de clúster quedó de la siguiente forma:

- Una instancia de Pgpool primaria.
- Una instancia de Pgpool de respaldo.
- Dos nodos PostgreSQL.

En cada nodo PostgreSQL se instaló Heartbeat en su versión 3.0.4 que es el software encargado de manejar la alta disponibilidad en el clúster tanto de los nodos PostgreSQL como de las instancias de Pgpool-II.

2.8 Incluir un sistema de almacenamiento compartido en la solución de clúster.

A la hora de implementar la alta disponibilidad de los nodos PostgreSQL se hace necesario de que estos tengan su clúster de datos en un almacenamiento compartido para en caso de que cualquiera de los dos caiga el nodo de respaldo pueda acceder a sus datos. Esto se hace necesario ya que como se está utilizando el particionado de datos, estos no son iguales en cada nodo PostgreSQL.

Para implementar dicha alternativa se utilizan los software iscsitarget y open-iscsi para el servidor y los clientes respectivamente. En el servidor van a estar almacenados los discos duros o particiones de estos a compartir entre los clientes, dichos clientes se conectan mediante la red hacia el servidor.

2.9 Conclusiones del capítulo.

El diseño de la solución se realizó teniendo en cuenta las características de los posibles entornos en los cuales se va a desplegar dicho clúster.

El diseño será desplegado en un escenario de pruebas en el laboratorio y se le aplicarán un grupo de pruebas para su validación. El diseño de la solución incluyó además algunas consideraciones para el despliegue del clúster en su entorno real así como requerimientos de seguridad y una propuesta de cómo se debe realizar el monitoreo del clúster.

3
CAPÍTULO
Diseño de las pruebas y análisis de los resultados

3.1. Descripción de las pruebas.

Para validar el correcto funcionamiento de un sistema informático se le debe realizar un proceso de pruebas para demostrar que dicho sistema está apto para su uso.

El ciclo de pruebas se realizó con el objetivo de demostrar que la solución funciona correctamente permitiendo su uso y una mejora considerable en los tiempos de respuestas del sistema. Los objetivos que se persiguen con dichas pruebas son los siguientes:

- Verificar que el sistema es funcional.
- Determinar los posibles puntos débiles del sistema.
- Establecer un nivel de calidad del sistema.

En el proceso de pruebas se analizó el comportamiento de las funcionalidades de la solución para comprobar que se desempeña como se previó en el diseño. Las funcionalidades que se probaron son las que definen el funcionamiento de la solución en general y pueden afectar el correcto funcionamiento de la misma.

En el caso particular de esta solución para comprobar el funcionamiento correcto del clúster y medir su rendimiento se decidió realizar un conjunto de pruebas las cuales se clasifican en dos tipos:

Pruebas de configuración: se estudia el comportamiento de las funcionalidades de la solución para ver si se desempeña como se previó en el diseño de la variante de la propuesta de solución. Las funcionalidades de la solución que se probaron fueron las que definen el comportamiento del clúster y pueden afectar el correcto funcionamiento de la solución.

Pruebas de carga: se realizan con el objetivo de validar y evaluar la aceptabilidad de los límites operacionales del sistema bajo distintos volúmenes de trabajo mientras la solución de clúster mantiene la misma cantidad de nodos. Los objetivos que se persiguen con estas pruebas son los siguientes:

- Evaluar los resultados obtenidos contra los criterios de rendimiento.
- Encontrar la(s) fuente(s) de los problemas de rendimiento.
- Buscar niveles de rendimiento.

Para el caso particular de la realización de pruebas a los sistemas de clúster de bases de datos se han establecido distintas métricas a recolectar como son:

- Tiempo de respuesta promedio de las consultas.
- Cantidad de transacciones por unidad de tiempo.
- Porciento de uso de *CPU* (Unidad Central de Procesamiento).
- Porciento de uso de memoria *RAM* (Memoria de Acceso Aleatorio).
- Tráfico de red.

Aunque se tienen en cuenta todas las métricas anteriormente mencionadas se le presta mayor atención al tiempo de respuesta promedio de las consultas por lo que establece como métrica principal.

Con la realización de las pruebas de carga se obtienen los siguientes beneficios:

- Ayudan a recopilar datos de vital importancia para la planificación de la escalabilidad y capacidad del sistema.
- Ayudan a evaluar la idoneidad del particionado de datos.

En la investigación se realizará fundamentalmente una de las variantes de pruebas de carga. Se hará aumentando el nivel de concurrencia y la complejidad de las consultas manteniendo un número fijo de nodos para así poder determinar cómo se comportan los tiempos de respuesta de la solución.

Para generar carga en los sistemas sobre los que se ejecuten las aplicaciones que se utilizaron en el proceso de pruebas se realizó mediante el acceso simultáneo de un determinado número de usuarios. Para simular los usuarios concurrentes accediendo a la aplicación se utilizó la herramienta JMeter.

Esta herramienta permite grabar una navegación de un usuario por una aplicación y diseñar un plan de pruebas con ella, para poder repetirla y simular el número de usuarios concurrentes que se desee mostrando los resultados de las pruebas en una amplia variedad de informes y gráficas. Además facilita a una rápida detección de los cuellos de botella existentes debido al tiempo de respuesta excesivo.

Tener en cuenta que esta prueba se realizará sobre computadoras personales con diferentes niveles de recursos que los servidores reales en los que será desplegada la aplicación. Esto trae consigo que los resultados obtenidos no sean exactos pero ayudan a determinar como mejoran o se deterioran las métricas establecidas.

3.2. Funcionalidades del sistema a probar:

- Replicación.
- Particionado de datos.
- Alta disponibilidad en el nodo Pgpool-II principal.
- Recuperación ante fallos de los servidores de bases de datos.
- Correcto funcionamiento de la aplicación.

3.3. Comportamiento esperado de las funcionalidades del sistema a probar:

Replicación: se espera que cuando se realice alguna operación de modificación de alguna base de datos/tabla replicada, la misma se realice sobre todos los servidores de bases de datos mediante el proceso de replicación.

Particionado de datos: se espera que al realizar alguna operación de inserción o selección de datos de las bases de datos particionadas esta sea ejecutada en los servidores correspondientes donde se correspondan los datos.

Alta disponibilidad en el nodo principal Pgpool-II: se espera que si el distribuidor primario deja de prestar sus servicios el distribuidor de respaldo sea capaz de recuperar el sistema automáticamente y sin la intervención del administrador.

Recuperación ante fallos de los servidores de bases de datos: se espera que si uno de los servidores de bases de datos deja de prestar sus servicios el software de alta disponibilidad lo detecte y mande a iniciar el nodo de respaldo tomando este el control.

Funcionamiento correcto de la solución: se espera que el comportamiento de la aplicación al ejecutarse sobre la propuesta de solución diseñada para ella y sobre el despliegue original sea el mismo, es decir que la existencia del clúster sea transparente al usuario.

3.4. Entorno de Prueba.

La configuración del entorno donde se vayan a ejecutar las diferentes pruebas que se le realizan a un software es un aspecto muy importante dentro del proceso de pruebas, pues si no se analizan bien los recursos de software y hardware que necesita el producto que se está construyendo, a la hora de probar dicho producto se prescindirá de los elementos necesarios para la ejecución de un proceso de pruebas exitoso.

Para ello se tuvo en cuenta a la hora de llevar a cabo todo el proceso de pruebas al sistema algunos requerimientos de hardware y de software, en aras de que las mismas lograran minimizar los errores en la aplicación, aunque vale aclarar que no son los óptimos que se necesitan para el mejor funcionamiento de la solución. A continuación se hace mención a algunos de los requerimientos que se consideran necesarios:

Hardware utilizado en los servidores:

- 256 Mb de memoria RAM
- Mcroprocesador Intel(R) Core(TM)2 Duo CPU E4500 @ 2.20GHz
- 80 Gb de disco duro.
- Conexión de red TCP/IP 100 Mb/s.

Sottware utilizado en los servidores:

- Debian GNU/Linux 6 (Squeeze)
- PostgreSQL 9.0.4.
- Pgpool-II 3.0.3.
- Heartbeat 3.0.4.

3.5 Pruebas al diseño genérico de la propuesta de solución.

Para validar que el diseño se desempeña correctamente se realizaron pruebas de configuración a las principales funcionalidades del clúster. El listado de las pruebas realizadas y el propósito de cada una de estas se describe en la tabla 1.1.

DG-01	Configuración	Comprobar que el diseño genérico de la propuesta de solución realiza la replicación correctamente.
DG-02	Configuración	Comprobar que el diseño genérico de la propuesta de solución realiza la distribución de los datos correctamente.
DG-03	Configuración	Comprobar que el diseño genérico de la propuesta de solución es capaz de recuperase ante fallos del nodo pgpool primario.
DG-04	Configuración	Comprobar que el diseño genérico de la propuesta de solución es capaz de recuperase ante fallos de un nodo PostgreSQL.
DG-05	Carga	Mostrar qué sucede con el rendimiento a medida que se va aumentando las peticiones.

Se realizó el despliegue de la solución propuesta siguiendo las indicaciones del documento “Guía de instalación y configuración de un clúster de alto rendimiento y alta disponibilidad para sistemas que gestionan grandes volúmenes de datos utilizando PostgreSQL”, el cual está compuesto por un nodo distribuidor de datos primario, uno de respaldo y dos servidores de datos.

Prueba de configuración DG-01: para analizar si se realiza la replicación de los datos se van a realizar operaciones de modificación de los mismos, y posteriormente se analizará si las consultas que se realizaron, si se hicieron a su vez sobre todos los servidores de bases de datos, esta información se va a extraer de los logs de los servidores PostgreSQL.

Prueba de configuración DG-02: para analizar cómo se comporta el proceso de distribución de los datos se van a realizar operaciones de modificación de los mismos y posteriormente se analizará si las consultas que se realizaron, se hicieron sobre los nodos correspondientes de acuerdo a las funciones de distribución definidas, esta información se va a extraer de los logs de de los servidores PostgreSQL.

Prueba de configuración DG-03: para simular la caída del nodo primario se apagó el nodo, observando el comportamiento del nodo de respaldo se pudo ver cómo la herramienta de alta disponibilidad detectó que el nodo primario estaba fuera de servicio y pasó todos los recursos hacia el nodo de respaldo, siendo esto transparente al usuario.

Prueba de configuración DG-04: para simular la caída de un nodo PostgreSQL se apagó uno de los servidores. Observando el comportamiento del nodo PostgreSQL de respaldo se pudo ver cómo la herramienta de alta disponibilidad detectó que el nodo PostgreSQL primario estaba fuera de servicio y pasó todos los recursos hacia el nodo de respaldo, siendo esto transparente al usuario.

Prueba de carga DG-05: para desarrollar las pruebas de carga a la solución, inicialmente se montó un servidor PostgreSQL fuera de la solución, en el cual mediante la herramienta JMeter, se le simuló 200 usuarios conectados de manera concurrente para así observar su rendimiento.

Posteriormente se le aplicó la misma prueba al clúster conformado por dos servidores PostgreSQL, aumentando el rendimiento del mismo, ya que para el servidor solo de PostgreSQL se comete un error de un 10 % al recibir dicha cantidad de usuarios conectados concurrentemente.

3.6. Análisis de los resultados.

Durante las pruebas de configuración realizadas al diseño genérico del clúster de alto rendimiento y alta disponibilidad se probaron las funcionalidades: replicación de datos, particionado de datos y recuperación ante fallos de los nodos pgpool y PostgreSQL.

En las pruebas de carga que fueron aplicadas a la solución se observó una mejora dado que el rendimiento aumenta en comparación con un solo servidor PostgreSQL, además de una eliminación de los errores. A continuación se mostrarán los resultados obtenidos de las principales métricas en dichas pruebas.

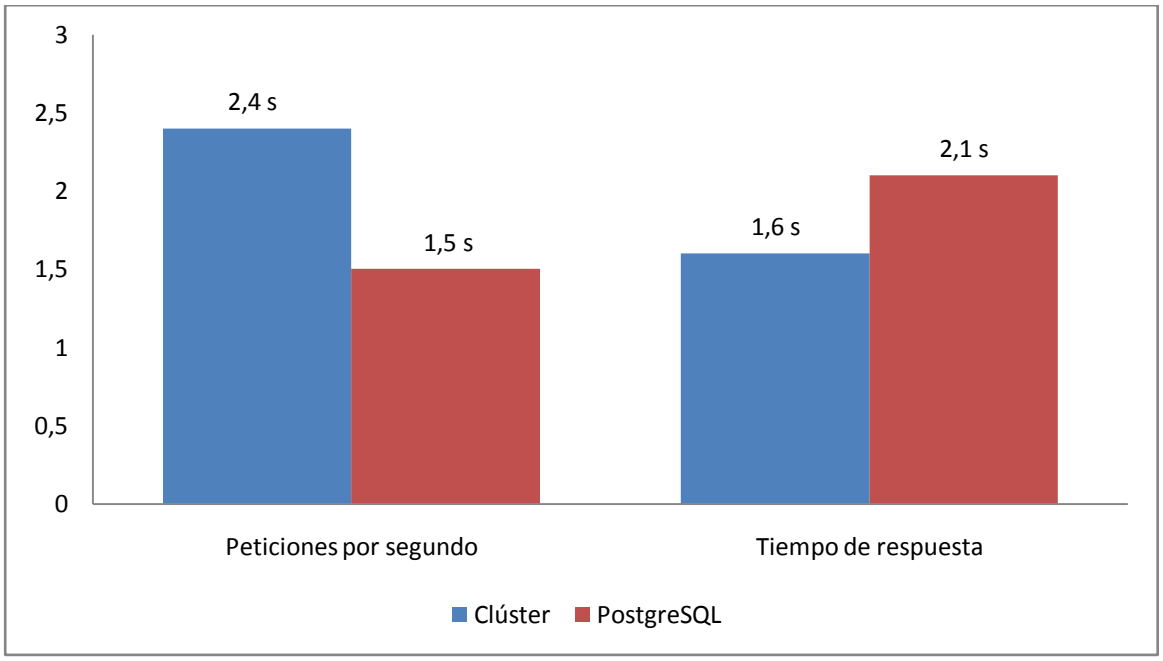

Figura 12: Peticiones por segundo y tiempo de respuesta.

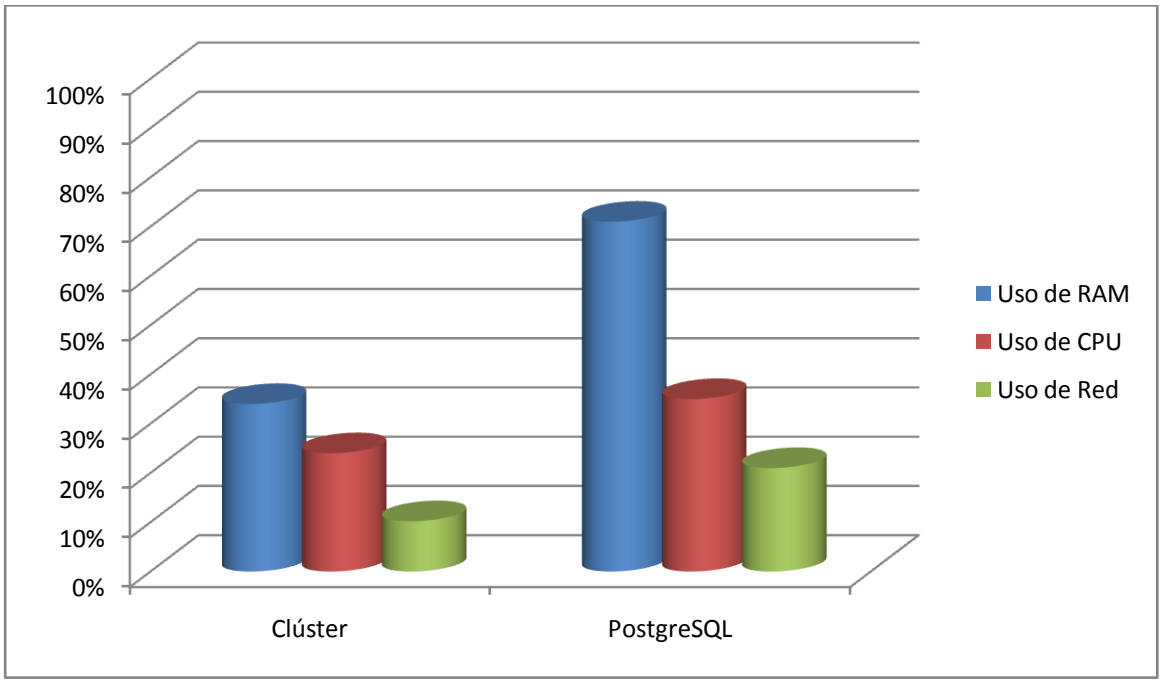

Figura 13: Consumo de recursos.

Como el comportamiento obtenido para cada una de las funcionalidades probadas fue satisfactorio se puede concluir que: el diseño del clúster de alto rendimiento y alta disponibilidad genérico está validado funcionalmente.

3.7. Conclusiones del capítulo.

Las pruebas se realizaron para validar la propuesta de solución diseñada anteriormente. Dicha solución fue sometida a dos tipos de pruebas: pruebas de configuración y pruebas de carga. Las pruebas de configuración se hicieron con el objetivo de comprobar que la solución de clúster no afecta el funcionamiento de las aplicaciones, además se comprobó el funcionamiento correcto del sistema. Las pruebas de carga se realizaron principalmente para medir los tiempos de respuesta una vez que las aplicaciones se encontraran desplegadas sobre las soluciones de clúster. Además se realizó un análisis del consumo de recursos de los nodos integrantes del clúster.

CONCLUSIONES GENERALES

A partir de la investigación realizada, que constituye un aporte a las tecnologías de bases de datos, orientada a las que gestionen grandes volúmenes de datos, se concluye:

1. Se obtuvo el diseño de una propuesta de solución que permite aumentar la capacidad de respuesta y la disponibilidad de sistemas que utilicen grandes volúmenes de datos.
2. La implementación de la solución basada en un clúster de alta disponibilidad y alto rendimiento, se realizaría utilizando la combinación de herramientas libres como lo son Pgpool-II para distribuir y replicar los datos y Heartbeat para implementar la alta disponibilidad.
3. Se validó la implementación de dicha propuesta de diseño teniendo en cuenta los tiempos de respuestas obtenidos durante la variación de la concurrencia de usuarios.

RECOMENDACIONES

Una vez cumplido el objetivo general de la investigación, se recomienda:

1- Seguir el estudio de las posibles reglas de distribución de datos para obtener un mejor rendimiento en las consultas.

BIBLIOGRAFÍA

Almaguer, Chávez, Dayrel and García, Riera, Dysán. 2008. *Cluster de servidores de bases de datos para aplicaciones web, sobre software libre.* 2008.

Andreasson, O. 2010. Iptables tutorial. [En línea] 2010. http://www.frozentux.net/documents/iptables-tutorial.

Apache Software Foundation, JMeter. 2010. The Apache Jakarta Proyect. [En línea] 2010. http://jakarta.apache.org/jmeter.

Barroso, L.,A. 2010. *Web search for a planet: The Google cluster.* s.l. : IEEE, 2010.

BLSF Equipo de desarrollo, doc. 2010. *Más Allá de Linux.* s.l. : Microsoft Corporation, 2010.

Bourke, T.",. 2010. *Server Load Balancing.* s.l. : O'Reilly & Associates, 2010.

CAVSI. 2010. CAVSI. [En línea] 2010. http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.

Clavijo, Paulo. 2010. Clusters de Alta Disponibilidad (HA). [En línea] 2010. http://www.lintips.com/?q=node/119..

Ernesto, Quiñones. 2010. *Introducción a PostgreSQL.* s.l. : APESOL, 2010.

Freedman, A. 2010. [En línea] 2010. http://www.agapea.com/Diccionario-bilingue-de-Computacion-n9673i.htm.

Galstad, E. 2010. Nagios Version 3.x Documentation. [En línea] 2010. http://www.nagios.org.

Goard. SysStat Documentation. [En línea] http://pagesperso-orange.fr/sebastien.godard/documentation.html.

Meier, J.,D. 2010. *Performance Testing Guidance for Web Applications.* s.l. : Microsoft Corporation, 2010.

Oracle. 2010. Oracle. [En línea] 2010. http://www.oracle.com/es/products/database/options/rac/index.html.

Pgpool Home Page, Documentation. 2010. Pgpool Home Page. [En línea] 2010. http://pgpool.projects.postgresql.org/doc.

PL/Proxy Project Home Page, Documentation. 2010. PL/Proxy Project Home Page,Documentation. [En línea] 2010. http://plproxy.projects.postgresql.org/doc.

PostgreSQL Development Group, PostgreSQL. 2010. PostgreSQL 9.0.1 Documentation. [En línea] 2010. http://www.postgresql.org/docs/9.0/interactive/pgbench.html.

Puentes, Samuel Hernández. 2010. Universidad de La Habana. [En línea] 2010. http://fbio.uh.cu/sites/bioinfo/glosario.html.

Rational Unified Process, Concepts. 2010. Concepts: Types of Test. [En línea] 2010. http://rup.hops-fp6.org/process/workflow/test/co_tytst.htm.

Sabater, Jaume. 2010. Replicación y alta disponibilidad de PostgreSQL con pgpool-II. [En línea] 2010. http://linuxsilo.net/articles/postgresql-pgpool.html..

Suárez, José. 2010. *Sistema HA bajo Linux.* 2010.

ANEXOS

Guía de instalación y configuración de un clúster de alto rendimiento y alta disponibilidad para sistemas que gestionan grandes volúmenes de datos utilizando PostgreSQL.

GLOSARIO DE TÉRMINOS

ACID (Atomicity, Consistency, Isolation and Durability): Atomicidad, Consistencia, Aislamiento y Durabilidad.

API: Application Programming Interface.

BSD: Berkeley Software Distribution.

Clúster: conjunto de computadoras, a menudo con semejantes componentes de hardware que se interconectan entre sí a través de un sistema de red de alta velocidad y son capaces de elevar la eficiencia para realizar determinadas tareas que individualmente no podrían realizar debido a la creciente necesidad de potencia computacional que demandan algunas aplicaciones.

Cron: administrador de tareas de Linux.

DATEC: Centro de Tecnología de Datos.

Erlang: Lenguaje diseñado para desarrollar sistemas altamente concurrentes y tolerantes a fallos.

HA: Alta Disponibilidad.

Linux Virtual Server (**LVS**): solución para gestionar balanceo de carga en sistemas Linux, su principal objetivo es desarrollar un servidor Linux de alto rendimiento que proporcione buena escalabilidad, confiabilidad y robustez usando tecnología de clúster.

Log: registro oficial de eventos durante un rango de tiempo en particular.

MVCC: Control concurrente multiversión.

PITR: Punto en tiempo de recuperación.

RAC: Real Application Clúster.

RAM: Random Access Memory, tipo de memoria de ordenador a la que se puede acceder aleatoriamente.

SGBD: Sistema Gestor de Bases de Datos.

SSH: Secure Shell, protocolo informático que sirve para acceder a máquinas remotas.

Supercomputadoras: ordenador con capacidades de cálculo muy superiores a las comunes. **High Performance**: Alto rendimiento.

TAF: Recuperación transparente ante fallas.

UCI: Universidad de las Ciencias Informáticas.