# UNIVERSIDAD NACIONAL DE CAJAMARCA
# FACULTAD DE INGENIERÍA

## ESCUELA ACADÉMICO PROFESIONAL
## DE INGENIERÍA DE SISTEMAS


# CLÚSTER DE SERVIDORES LINUX PARA ALTA DISPONIBILIDAD DE LA INFORMACION

TESIS
PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

BACHILLER. Lester Juan De Dios Villar Zamora.

ASESORES: MCs. Ing. Carlos Jesús Koo Labrin.
MCs. Lic. Víctor Sánchez Cáceres.

Cajamarca, Perú

Noviembre - 2014

## DEDICATORIA

A ti que me diste todo, sin pedir nada

A ti que dejaste todo por mí

A ti que entregaste todo por mí...

Gracias Mamá por hacerme

Comprender que el amor verdadero es ese que se entrega,

Sin esperar nada a cambio.

## AGRADECIMIENTO

A DIOS fuente inagotable de amor y conocimiento, a mis Padres Norma y Tito, hermanos Raúl, Rodrigo y Martin, tíos Magno y Cristina y a mi novia Claudia Sofía por su constante apoyo para poder lograr este objetivo en mi vida...

# CONTENIDO

# INDICE DE FIGURAS

# INDICE DE TABLAS

# INDICE DE GRÁFICOS

## RESUMEN

La investigación realizada tiene como objetivo implementar y diseñar un clúster de servidores en entorno Linux para brindar un servicio de alta disponibilidad de la información, para tal propósito se utilizaron herramientas como la base de datos postgreSQL, Pppool II, Heartbeat. La investigación tiene como caso de estudios la empresa MINE SENSE SOLUTIONS que implementará un sistema de control de flotas pesadas para el proyecto minero Hudbay-Constancia en la región del Cusco. La empresa en cuestión tiene la necesidad de alta disponibilidad del servicio de información para el sistema que implementará por ser de vital importancia en las operaciones diarias. Para el diseño e implementación del clúster de alta disponibilidad se investigó las herramientas disponibles y que mejor se adaptan a las necesidades antes mencionadas.

La implementación comienza con la instalación del SO Ubuntu Server, luego se procede con la configuración de los IP estáticos necesarios para el tráfico de información y la configuración inicial de PostgreSQL para el acceso con el usuario root por defecto, se necesita acceso del tipo SSH entre los servidores para lo cual es necesario crear claves públicas para el acceso remoto desde los diferentes nodos que conforman el cluster. Luego de la configuración SSH se tendrá que configurar la replicación de información para lo cual se utilizó el mecanismo de espejo del motor de Base de Datos PostgreSQL, Stream Replication. Al tener los nodos en espejo ya es posible manipular los roles de los nodos a través del administrador del clúster, el middleware pgpool-II y después de la configuración del middleware es necesario configurar la alta disponibilidad del controlador del clúster a través de la herramienta por excelencia en Linux para alta disponibilidad, Heartbeat. Los resultados obtenidos luego de la implementación del clúster son: alto rendimiento del servicio de datos, escalabilidad en la arquitectura del clúster, balanceo de carga entre los nodos del clúster los cuales se distribuyen el tráfico de información y las transacciones requeridas, haciendo así que se aminore el procesamiento en los nodos del clúster. Al tener un alto rendimiento respecto a la continuidad ante caída de nodos (failover) se pudo reducir el tiempo de inoperatividad estimado llegando a tener el 99.99 de disponibilidad de la información en un año.



**Palabras clave:** clúster, disponibilidad, servidor, información, rendimiento, escalabilidad, balanceo de carga, middleware, base de datos, nodo, espejo, failover, script bash.

## ABSTRACT

The research conducted aims to implement and design a server cluster in Linux environment to provide high service availability of information, for the purpose tools such as data base postgreSQL, Pppool II, Heartbeat is used. The research is case study MINE SENSE SOLUTIONS Company to implement a control system for heavy fleets Hudbay-Constancia mining project in the Cusco region. The company in question is the need for high service availability information for the system being implemented vital in daily operations. For the design and implementation of high availability cluster tools available investigated and best meet the above requirements.

Implementation begins with the installation of the OS Ubuntu Server, then proceeds with configuring static IP required for traffic information and initial configuration of PostgreSQL for access to the root user by default, type access SSH is needed between servers for which it is necessary to create public keys for remote access from different nodes in the cluster. After configuring SSH will have to configure replication of information for which the mirror mechanism motor PostgreSQL Database, Stream Replication was used. By having nodes mirror is possible to manipulate the roles of the nodes through cluster administrator, middleware pgpool-II and after middleware configuration is necessary to configure high availability cluster controller through the tool excellence in Linux for high availability, Heartbeat. The results obtained after implementation of the cluster are: high performance data service, scalable cluster architecture, load balancing across the cluster nodes where traffic information and the required transactions are distributed, thus making it lessen the processing cluster nodes. To have high performance with regard to continuity drop nodes (failover) was reduced inoperability estimated time 99.99 coming to have availability information in a year.



**Key words:** cluster, availability, server information, performance, scalability, load balancing, middleware, database, node, mirror, failover, bash script.

# CAPÍTULO I. INTRODUCCIÓN

En la actualidad existen grandes sistemas informáticos con gran demanda de información que necesitan funcionar en forma ininterrumpida y sin errores los 365 días del año ya que el servicio que brindan estos es crítico y de vital importancia. El éxito o declive de una empresa u organización depende de múltiples factores tales como el capital el recurso humano la energía, etc., pero en la actualidad también depende de cómo es que aprovecha sus activos intangibles tales como la información que hoy en día resulta ser tal vez el activo más importante que tienen las empresas u organizaciones ya que su mala gestión puede repercutir de manera directa en la continuidad de la organización. El fallo de los grandes sistemas informáticos debido a fallos en el suministro eléctrico, las comunicaciones, el hardware o el software, incendios, robos, inundaciones, ataques informáticos internos o externos, y amenazas por software malicioso o virus, tiene como consecuencia un gran margen de tiempos de indisponibilidad que trasladados a cifras monetarias implican grandes pérdidas para diversas empresas y organizaciones. Esta investigación tiene como caso de estudios la empresa MINE SENSE SOLUTIONS que implementara un sistema de control de flotas pesadas para el proyecto minero Hudbay-Constancia en la región del Cusco, la empresa en cuestión tiene la necesidad de alta disponibilidad del servicio de información para el sistema que implementara ya que es el sistema de vital importancia en las operaciones diarias, ante este escenario surge la interrogante: ¿Cómo se podrá implementar un Clúster de Servidores para obtener Alta Disponibilidad de la información?.

Con la implementación de un clúster de servidores se podrá obtener Alta Disponibilidad de la información en el sistema implementado por la empresa MINE SENSE SOLUTIONS SAC, para el proyecto minero Hudbay-Constancia. Para concecucion de esta afirmación es necesario cumplir con los tres objetivos, caracterizar el estado actual y las necesidades de la disponibilidad de la información en la Empresa MINE SENSE SOLUTIONS SAC, diseñar un clúster de

servidores para obtener Alta Disponibilidad de la Información., validar el diseño propuesto que asegure la Alta Disponibilidad requerida.

La presente investigación cuenta con cinco capítulos, en capitulo II se hace hincapié a los antecedentes teóricos que existen en la actualidad referentes al tema de la investigación, también se hace referencia a los bases teóricas en las cuales se fundamenta la investigación tales como clúster, Alta Disponibilidad y clúster de alta disponibilidad, por último en este capítulo se muestra una definición de términos, los cuales son utilizados en la investigación y con el propósito del mejor entendimiento de la misma se describe a cada uno de estos.

En el capítulo III se realiza el desarrollo metodológico en el cual se ve el estado actual y necesidad de alta disponibilidad de la información, el diseño del clúster y su implementación, también se desarrolla el análisis, tratamiento de datos y presentación de resultados para el pre test (antes del diseño e implementación del clúster) y el post test (después del diseño e implementación del clúster).

En el capítulo IV se ve el análisis y discusión de resultados al comparar el pre test con el post test y con la ayuda de herramientas estadísticas aceptamos o rechazamos las hipótesis planteadas de acuerdo a cada uno de los indicadores, luego se realiza un análisis de resultados en el cual se describe a detalle lo que se obtuvo en las mediciones para cada uno de los indicadores propuestos.

En el capítulo V se plasman las conclusiones y recomendaciones de la investigación de forma clara y concisa.

# CAPÍTULO II. MARCO TEÓRICO

## 2.1 Antecedentes teóricos de la investigación.

### 2.1.1 Configuración de un clúster de alta disponibilidad y balanceo de carga en Linux para satisfacer gran demanda web y servicios de resolución de nombres.

La presente investigación analiza una técnica para brindar alta disponibilidad por medio de la denominada redundancia mediante la utilización de varios servidores instalados en lugar de uno sólo, los cuales tengan la capacidad de trabajar en paralelo y de asumir las caídas de algunos de sus compañeros; por otra parte esta técnica permitirá la adición y eliminación de servidores al grupo según las necesidades. A esta técnica se la denomina clústers de servidores (granja de servidores).

La investigación se encuentra destinada también al análisis del balanceo de carga, con lo cual surge el concepto de "clúster de servidores virtuales", el cual permite que un conjunto de servidores de red (clúster) compartan la carga de trabajo y el tráfico de diferentes clientes, haciendo que el servicio sea transparente para el cliente dando la ilusión de que es único servidor. Al balancear la carga de trabajo en un conjunto de servidores se mejora el tiempo de acceso y la confiabilidad. Además como es un conjunto de servidores el que atiende el trabajo la caída de uno de ellos no ocasiona una caída total del sistema. Este tipo de servicio es de gran valor para compañías que trabajan con grandes volúmenes de tráfico y trabajo en sus servidores: Web, correo electrónico y bases de datos.

Los objetivos de la presente investigación se desarrollan a lo largo de cuatro capítulos, en el primer capítulo se realiza un breve estudio de las tecnologías paralelas, tecnología clúster, sus componentes de hardware y software, características, clasificación, y arquitectura de funcionamiento. Además un breve

estudio relacionado a la administración y planificación de tareas. El segundo capítulo se realiza un breve estudio de los sistemas de archivos existentes, estudio de herramientas para implementar clúster de alta disponibilidad y balanceo de carga, en especial HA OSCAR y LVS respectivamente, además se mencionan algunas herramientas de administración y planificación de tareas.

En el tercer capítulo se realiza un breve estudio de otras implementaciones realizadas, se especifican los requerimientos de hardware y software necesarios para la configuración del clúster, se configuró el clúster con las herramientas de código abierto como son HA OSCAR (Open Source Cluster Aplication Resources High Avaibility) y LVS (Linux Virtual Server) para alta disponibilidad y balanceo de carga respectivamente, se diseñó la aplicación Web e implementación de servicios (resolución de nombres y servicio de correo electrónico - Postfix). Una vez concluida la fase de instalación y configuración, se realizan las pruebas del caso utilizando el benchmark ECperf, el cual es el encargado de generar un alto tráfico http con la finalidad de verificar el funcionamiento real del clúster configurado, para finalizar el capítulo se compara los resultados obtenidos con otras tecnologías de servidores Web, correo electrónico, de resolución de nombres, tecnologías SAN (Storage Area Network) y NAS (Network Attached Storage).

Las conclusiones de la investigación fueron las siguientes:

Una vez realizado el estudio de la tecnología clúster se puede concluir la importancia que tiene el software libre en su configuración e instalación, las herramientas y aplicativos que el sistema operativo Linux ofrece lo hacen más atractivo, en especial la posibilidad de manipular su kernel para añadir o hacer uso de los diferentes paquetes que se requieren para el funcionamiento de aplicativos y servicios que se este brindando, así como la conveniencia en cuanto a costo ya que utilizar otras herramientas propietarias involucra un gasto adicional.

Después de estudiar la tecnología clustering se puede mencionar la importancia que tiene en ambientes donde se requiere un incremento del número de transacciones o velocidad de respuesta, ofrecido por el clúster de balanceo de carga, incremento de confiabilidad y robustez ofrecido por el clúster de alta disponibilidad. Estas características son de fundamental importancia en el funcionamiento de un sistema de gran demanda Web, donde el rápido crecimiento

del Internet, incremento de comercio electrónico, y otras aplicaciones Web requieren un adecuado tiempo de respuesta y en especial una adecuada disponibilidad de los servicio ofrecidos, estos requerimientos son muy importantes para los usuarios y empresas que ofrecen estos servicios, ya que la indisponibilidad y lentitud en el sistema implicaría pérdidas económicas considerables.

Una de las principales ventajas de tener un clúster es la escalabilidad que éste ofrece, ya que dependiendo de las necesidades de procesamiento permite incrementar servidores reales sin tener un límite establecido, ofreciendo una gran ventaja en comparación a otras tecnologías como supercomputadores, que poseen un gran funcionamiento pero cuando las necesidades incrementan se encuentran limitados debido a que saturan su capacidad de cálculo y para lograr escalabilidad se debe cambiarlo por uno más potente. La escalabilidad en cuanto a costo es más conveniente cuando se tiene un clúster con máquinas personales y software libre, ya que es más conveniente invertir en un servidor de estas características que en un supercomputador el cual es más costoso.

La alta disponibilidad en un ambiente de gran demanda Web y resolución de nombres tiene mucha importancia en diferentes aspectos, uno de ellos es la facilidad de realizar el mantenimiento de servidores. Una máquina de un clúster de servidores se puede sacar de línea, apagarse y actualizarse o repararse sin comprometer los servicios que brinda el clúster y sin afectar el desempeño del sistema en general.

Los mecanismos de balanceo de carga existentes ofrecen un adecuado funcionamiento, pero es muy importante tomar en cuenta diversos aspectos antes de escoger cual de ellos utilizar. En el presente caso de estudio se decidió utilizar el mecanismo de enrutamiento directo, debido a que las características de hardware del nodo director primario y standbye no eran las adecuadas para realizar traslación de direcciones (balanceo por NAT) y encapsulamiento IP, los cuales con un alto tráfico necesitan gran capacidad de cálculo para poder trabajar de manera similar al utilizado por enrutamiento directo.

De la pruebas realizadas en el clúster de balanceo de carga se pudo constatar que la granja de servidores ubicada tras lo nodos directores ofrecen gran desempeño al momento de dar un determinado servicio, ya que por el hecho de

existir varios servidores reales configurados de idéntica manera permiten tener un mayor número de conexiones simultáneas, y con ello tener mayor eficiencia de funcionamiento del clúster, la diferencia de funcionamiento se hizo notoria cuando se realizó las pruebas con dos servidores reales configurados pero uno de ellos inactivo, el número de conexiones en un tiempo determinado disminuyó y el tiempo de respuesta aumentó con relación al grupo de servidores reales ofreciendo el mismo servicio.[1]

### 2.1.2 Alta disponibilidad en servidores y optimización de recursos hardware a bajo coste.

La presente investigación esta evocada al estudio, diseño, implementación real y experimentación de un sistema de Alta disponibilidad para servidores virtualizados, con el que pretendemos dotar a los sistemas que implantemos de independencia del hardware sobre el que se ejecuta y a su vez de unos tiempos de recuperación e indisponibilidad reducidos. Todo ello con una reducción de costes importante en comparación con los sistemas comerciales existentes, reduciendo dichos costes tanto en su componente hardware como software, mejorando la flexibilidad del sistema.

Además se realizó un estudio exhaustivo de las tecnologías para la implementación de los diferentes componentes del mismo, comparándolas muchas de ellas de forma práctica identificando la mejor combinación en cada caso de cara al sistema completo propuesto y considerando diferentes ámbitos de aplicación.

Uno de los principales objetivos del presente trabajo de investigación es conseguir un sistema que nos permita ofrecer alta disponibilidad en servidores y un mejor aprovechamiento de recursos hardware a bajo coste, para que cualquier empresa y/o servicio, ya sea con pequeño o mediano presupuesto, pueda disponer de este tipo de configuraciones y dotar a sus servicios de una mejor disponibilidad, una menor dependencia del hardware del que dispongan y un mejor aprovechamiento del mismo gracias a la virtualización.

No obstante, cuando hablamos de alta disponibilidad en servidores debemos tener presente que no es lo mismo que intentar ofrecer alta disponibilidad en servicios. En primer lugar, los tiempos para migrar un servicio caído en un servidor a otro nodo del clúster suelen ser inferiores a los tiempos para migrar un servidor completo, y por otro lado, si estamos monitorizando servidores, no tenemos por

qué darnos cuenta si un servicio que está corriendo en un determinado servidor está fallando o no, con lo que podríamos tener un servicio no disponible durante un tiempo indeterminado hasta que otras herramientas externas al clúster nos informasen de que estos servicios están fallando.

El principal objetivo del presente estudio es evaluar las prestaciones de los sistemas de almacenamiento en clústers. Dado que es una parte fundamental de los sistemas basados en clúster y que en nuestra propuesta toma mayor relevancia debido a que sobre este sistema van a almacenarse las imágenes de los servidores virtualizados que van a corren en el mismo y de su rendimiento va a depender los servidores virtualizados que podamos ejecutar y el rendimiento de los mismos. Por lo tanto va a ser importante estudiar a fondo el rendimiento de cada uno de estos sistemas de almacenamiento y los sistemas de ficheros a usar.

Como primera y principal conclusión de esta tesis de Master podemos destacar la propuesta, implementación y experimentación de un sistema de alta disponibilidad a bajo coste. Basándonos en la reutilización del hardware del que ya disponemos, todo ello mediante el uso de los sistemas de almacenamiento y plataformas de virtualización estudiados y con los que hemos experimentando durante la realización de este trabajo y se ha comprobado que aportan a la implementación final una estabilidad, fiabilidad y rendimiento más que aceptables para estos entornos y que detallaremos más adelante.

Todo ello nos va a permitir ofrecer a pequeñas, medianas y grandes empresas, que no puedan o quieran asignar excesivos recursos económicos a este fin, poder disponer de una serie de servicios/servidores disponibles sin depender completamente del hardware en el que estén ejecutándose y permitiendo la restauración de los mismos en otros equipos en caso de desastre inminente.

En un primer lugar se ha estudiado la posibilidad de disponer de un sistema de almacenamiento distribuido y accesible desde diferentes sistemas físicos así como contrastado diferentes sistemas de ficheros que nos dan la posibilidad de acceso concurrente a estos recursos. Se ha podido apreciar tras comparar costes y eficiencia que lo que más se adapta a lo que inicialmente estamos buscando es una combinación de hardware/software basado en la utilización de discos locales (los mismo que ya se suelen tener en los servidores de la empresa) y la tecnología/software DRBD que nos permitirá crear un raid 1 (mirroring) a través de la red, proporcionándonos seguridad y a la vez combinado con un sistema de

archivos de clúster tipo ocfs2 y/o gfs2, la posibilidad de acceder a los mismos desde dos nodos diferentes de nuestro clúster.

En segundo lugar se ha estudiado diferentes sistemas de virtualización para ejecutar en ellos los servidores/servicios que queramos ofrecer y ver cual se adapta mejor a nuestro propósito. En este punto se puede destacar como plataforma recomendada Xen, aunque en algunos casos, por otros motivos como la elección del sistema host o por el sistema de almacenamiento puede llegar a ser una mejor opción optar por plataformas como Qemu/KVM.

Por último, se ha estudiado diferentes sistemas de clustering que ofrezcan alta disponibilidad, descartando directamente los que necesitan una inversión elevada para llegar a obtener las prestaciones requeridas, como son las soluciones VMWare VSphere y XenServer. De las dos estudiadas se puede decir que son perfectamente válidas y que la elección entre una u otra puede venir dada más por elecciones externas como el sistema de ficheros de clúster o la predilección por una u otra herramienta de configuración.[2]

### 2.1.3 Solución de Alta Disponibilidad de base de datos por hardware o por software.

En el presente trabajo de investigación expone la alta disponibilidad de las bases de datos puesto que ya no se trata de una simple preferencia de las organizaciones, ahora es un factor crítico para su éxito, sobre todo, cuando la información vital de las empresas se comparte electrónicamente entre los empleados, socios y proveedores. La más mínima falla de la red puede resultar en la pérdida de ganancias, pérdida de productividad o algo peor.

Un clúster es una colección de computadores independientes que ejecutan una serie de aplicaciones de forma conjunta y aparecen ante clientes como un solo sistema. Los clústers son una solución de hardware al que se le puede agregar un almacenamiento redundante y éste ser una buena solución de alta disponibilidad de base de datos ya que el clúster puede ocultar ante los usuarios del sistema los fallos de componentes y proporcionar servicios de alta disponibilidad.

La replicación es una solución basada en software y es el proceso de mantener copias de los datos del sistema principal que pueden ser usados como datos alternativos sobre otros sistemas. Esas copias son usadas si el sistema principal necesita estar fuera de línea para efectuar copias de seguridad o rutinas de

mantenimiento (alta disponibilidad), en caso de emergencia, cuando el sistema principal haya fallado (recuperación de fallo) o en una situación de destrucción total como lo es un terremoto (recuperación de desastre). Una simple copia de seguridad restaurada en un sistema alterno, puede ser considerada como una forma de replicación.

Cuando se habla de alta disponibilidad en hardware, se hace referencia a tener dos máquinas con igual hardware y sistema operativo con un canal de alta velocidad y un ancho de banda grande (fibra óptica), la(s) máquina(s) comparte(n) el acceso a una base de datos en un arreglo de discos.

Cuando se habla de alta disponibilidad por software, se hace referencia a tener una máquina 1 con un tipo de hardware, sistema operativo y la base de datos del sistema situada en un punto geográficamente distante de otro local ya sea en unos cientos de kilómetros o bien en una parte del mundo, ésta utiliza un canal de comunicación con un ancho de banda normal o incluso por medio de Internet, con el cual se estará refrescando los cambios a la base de datos del lugar origen al remoto en donde se encuentra la máquina 2 con igual tipo de hardware, sistema operativo y una réplica de la base de datos de la máquina 1.

Es importante mencionar que para que las soluciones de alta disponibilidad de base de datos por hardware o software más comunes le ayuden a tener ideas, características y parámetros para la evaluación de las alternativas que mejor satisfagan las necesidades de la empresa que las consulta, es necesario que posea información sobre las soluciones más comunes sobre esta tecnología.

Las conclusiones de la investigación fueron las siguientes:

La clave de la alta disponibilidad de base de datos es la redundancia que permite mantener los datos en más de un lugar, con lo cual se logra en un momento dado la recuperación a un desastre. El tiempo fuera de servicio o downtime es categorizado como planeado y no planeado; indistintamente cual de los dos ocurra, debe ser minimizado para evitar cualquier rompimiento en el servicio, lo que implicaría pérdidas.

Los clústers son usados para ocultar las fallas en los nodos del sistema, sin dejar de prestar el servicio a los usuarios de las bases de datos y permiten distribuir la carga de trabajo, proporcionando un buen rendimiento en cualquier momento; además, permiten agregar nodos, lo cual los hace sobresalir en sus principales

ventajas que son la disponibilidad y la escalabilidad. La replicación es una solución de software que permite tener copias de la base de datos en producción en más de un lugar, ya sea una ubicación local o remota; estas copias actualizadas de las bases de datos de producción pueden ser usadas para reportes, consultas, Backus y en casos de emergencia cuando el sistema de producción haya fallado o esté sea totalmente destruido.

Las soluciones de alta disponibilidad de bases de datos por hardware o por software, al combinarlas proporcionan un mejor resultado. La solución de alta disponibilidad de base de datos por software Oracle Data Guard proporciona recuperación a desastres, si la base de datos está en una localidad remota; proporciona disponibilidad si se produce una falla en la base de datos primaria, o bien cuando se necesite operaciones de mantenimiento, permitiendo cambiar entre bases de datos primaria y standby.

La solución de alta disponibilidad de base de datos por software Oracle Real Application Clusters proporciona disponibilidad y escalabilidad debido a que está basada en clusters; proporciona recuperación a desastres si los nodos de la misma se encuentran a gran distancia entre ellos. La solución de alta disponibilidad de base de datos por software Shareplex para Oracle proporciona recuperación a desastres, manteniendo una base de datos en un lugar remoto constantemente actualizado por medio de la replicación, proporciona escalabilidad y disponibilidad ya que puede replicarse a más de un servidor. La solución de alta disponibilidad de base de datos por software SQL Server 2000 clúster proporciona disponibilidad y escalabilidad, debido a que usa tecnología de clusters y proporciona un buen rendimiento. No proporciona recuperación a desastres, ya que los nodos residen en el mismo lugar. La solución de alta disponibilidad de base de datos por hardware llamada RAID local, proporciona disponibilidad a través de distintos niveles de redundancia en los arreglos de discos, con lo cual se evita la pérdida de los datos si uno o más discos llegan a fallar; proporciona escalabilidad, la cual está limitada al chasis del arreglo de discos pero no proporciona recuperación a un desastre local.

La opción de almacenamiento SAN proporciona recuperación a desastre, si los sitios están copiados idénticamente y localizados a una distancia segura, debido a su tecnología de canal de fibra para proteger y distribuir datos; ofrece escalabilidad agregando almacenamiento, sin tener tiempo fuera de servicio;

reduce el tráfico en la red primaria. La opción de almacenamiento NAS comparte archivos a través de una red de área local; mejora el rendimiento, descargando el servicio de archivo proporcionado por el servidor. [3]

### 2.1.4 Método para el manejo del balanceo de carga en sistemas de computo distribuido de alto desempeño.

Esta investigación trabaja con las arquitecturas clúster de computadores, esto con el fin de acotar un poco más la problemática a tratar en el área de la computación paralela y distribuida, debido a su gran extensión. El enfoque principal se encontrará relacionado única y exclusivamente a una de las variables que afectan el rendimiento en los clúster: el balanceo de la carga de trabajo. El balanceo de carga trata de distribuir la carga de trabajo de acuerdo a la disponibilidad de procesamiento y a los recursos de cada equipo en un sistema computacional. Esta distribución pretende maximizar la utilización de los recursos, posibilitando el mejor desempeño del sistema[4].

En la actualidad el problema del balanceo de carga ha sido bastante trabajado, debido al hecho de que las diferentes comunidades requieren disminuir al máximo el tiempo de ejecución de las aplicaciones ejecutadas. Sin embargo, su solución no es siempre trivial. Este es un problema bastante complejo, debido a la dificultad en ocasiones de lograr que las propuestas en las distribuciones de carga de trabajo sean fácilmente escalables, o que se pueda trabajar con equipos que no sean necesariamente similares; en general se recurre a trabajar con clúster homogéneos (máquinas con software o hardware semejantes) y no en heterogéneos (máquinas con software o hardware diferentes).

El objetivo general de esta tesis es entonces, proponer un método para efectuar el manejo del balanceo de carga en sistemas de cómputo distribuido de alto desempeño, empleando técnicas propias de la computación paralela por medio del uso de un clúster heterogéneo. Los objetivos específicos de este trabajo son los siguientes, diferenciar entre los diversos tipos de clúster existentes con el fin de entender su funcionamiento básico para así verificar si es posible combinarlos, identificar las diferentes formas para balancear la carga entre nodos de un clúster, para escoger la mejor de estas técnicas y analizar el rendimiento en el sistema total, estudiar y seleccionar un sistema paralelo con el fin de modelarlo, basándose en uno de los modelos de computación y su determinada arquitectura

paralela, proponer un método para el manejo del balanceo de carga en sistemas de cómputo distribuido usando un clúster heterogéneo basado en el tipo de balanceo de carga seleccionado, la arquitectura y el modelo computacional, validar el método propuesto mediante la implementación de un prototipo con base en experimentaciones pasadas, realizando pruebas de desempeño.

En el marco de esta investigación se tiene como prioridad plantear este método de balanceo de carga siendo éste aporte valioso para la comunidad científica local. Aunque la parte de implementación es importante para la investigación ésta sólo alcanza un papel secundario para mostrar únicamente los resultados encontrados durante el proceso. La validación fue efectuada en un simulador de tiempo discreto llamado Network Simulator-2 (NS-2), en donde fue implementado el algoritmo de balanceo. Los nodos utilizados que conformaran el clúster son equipos que contarán únicamente con hardware diferente. La versión de NS-2 con la que se trabaja en esta investigación es la versión 2.31, instalada en Linux-SUSE versión 10.1 y ubicada en /usr/local/src/ns-2.31. El NS-2 se apoya en dos lenguajes de programación para su correcto funcionamiento: OTcl y C++. Como resultado de la simulación se obtiene una gran cantidad de datos matemáticos para los estudios posteriores y trazas específicas que son visualizadas en la herramienta NAM del ns. Los resultados obtenidos muestran que el método propuesto alcanza todas las expectativas logrando mejorar los tiempos de ejecución en comparación con los métodos con los que fue comparado.[5]

### 2.1.5 Implementación de un servidor web virtual balanceador de carga basado en Linux.

En la actualidad una tecnología predominante y necesaria es el Internet, el cual es usado principalmente para el envío y consulta de información mediante un navegador o explorador que nos facilita estas tareas, lo que conocemos como World Wide Web (WWW). Y para la disposición de la información es elemental que los sistemas de información trabajen al 100 %, pero al usuario no le interesa mucho este aspecto ya que lo que él necesita es consultar la información de uno u otro modo. El usuario no se da cuenta que publicar o tener la información disponible tiene un costo económico y de tiempo de procesamiento muy grande.

Una solución viable para que esto no llegara a interrumpir el proceso de envío y recepción de información sin tener necesidad de una inversión mayor es la

tecnología clustering. Ésta tecnología se puede comprender cuando nos cuestionamos ¿ cuánto tiempo necesita tener su sistema funcionando?. Es decir, qué tanto tiempo queremos que nuestra información esté disponible. La tecnología clustering, se refiere al hardware con lo cual se construirá lo que podemos describir como clúster.

En el presente trabajo de investigación, se analiza esta tecnología con la construcción de un clúster y una implementación real llamada LVS (Linux Virtual Server), con lo cual se demostrará que con pocos recursos económicos o ya existentes como puede ser un equipo obsoleto o un equipo de bajo costo se soluciona el problema de disponibilidad, escalabilidad y eficiencia que engloba un término llamado Alta Disponibilidad dentro de la WWW.

El objetivo principal de la presente tesis es implementar con Tecnología Clúster un servidor balanceador de carga tipo LVS haciendo uso de software libre y equipo de cómputo de limitadas características. Para implementar un servidor Web eficiente que funcione 24x7 los 365 días del año, se necesita considerar varios puntos importantes como: computadoras con buenas características, un lugar adecuado donde se deben instalarse las computadoras, una unidad de respaldo de energía, y personal idóneo para el mantenimiento del servidor.

Esto nos lleva a pensar que tan caro es poner un servidor y si realmente es justificable la inversión inicial, porque debemos de comprar una computadora con las últimas características como: son la velocidad de respuesta, almacenamiento primario, capacidad de presentar información en pantalla, entre otras cosas. Pero esto ya no es problema con la tecnología llamada clustering que implica construir un servidor con equipo homogéneo y no homogéneo que cumpla con las mismas funciones que una sola máquina. El equipo homogéneo y no homogéneo puede ser de diferentes características.

De aquí es importante demostrar los pros y contras de cómo se construye un clúster o un servidor balanceador de carga de Alto Rendimiento. El desarrollo del presente trabajo es dar a conocer cómo es un clúster LVS o de alta disponibilidad. El internet es una tecnología elemental para cualquier empresa. En años anteriores una empresa que necesitaba hacer un proceso administrativo para realizar una venta, un pedido e incluso reportes para la toma de decisiones, tardaba varios días, si la empresa es demasiado grande o fuera de la ciudad la

tendencia es implementar sistemas de información en una Intranet, o una WAN o Internet, para después emigrar a un Sitio WEB que ayude a agilizar estos trámites.

Pero la implementación de este tipo de sistemas de información no es tan sencilla, ya que deben de considerarse varios puntos importantes como son: equipo que se utilizará, lugar donde se instalará, proveedor de Internet, desarrolladores del sitio ( programadores, diseñadores, entre otros), así también debe de considerarse el mantenimiento del sitio, entre otros puntos. Pero un aspecto importantes es el tecnológico o hardware donde se instalará el sistema de información, que incluye: computadoras, impresoras, unidades de respaldo de energía (no-breaks), concentradores o switches, instalación eléctrica. Desde el punto de vista tecnológico, debemos de analizar qué tan moderno debe ser nuestro equipo de cómputo para que los costos de adquisición del mismo, no se incrementen tanto. Al implementar un sistema de información en el sistema, para tener ahorro de tiempo en la publicación de datos, puede ser más eficiente si la información se publicara en un sitio WEB para estar disponible en Internet. Esto tiene un costo elevado de la tecnología si se instala un moderno equipo de cómputo para la implementación del sitio WEB. Pero si las empresas no desean invertir mucho, y si desean tener lo más nuevo en vanguardia de Internet, aquí es donde la tecnología clustering se vuelve una buena alternativa. Este tipo de idea es lo que se conoce como clustering.

La investigación concluye: No es fácil construir un clúster debido a que los manuales no tienen todos los puntos a seguir para la construcción los cuáles omiten muchos pasos. Antes de construir algo se tiene que leer lo suficiente para poder empezar con una idea de cómo se construye. El clúster necesita un hardware idóneo, como pueden ser las tarjetas de red y el concentrador de mayor velocidad u otro tipo de conexión, para mejorar velocidad de respuesta y transferencia del clúster, otro tipo de dispositivo son los discos duros, la memoria RAM, entre otras cosas. La unidad de respaldo de energía, No break, es importante debido a que una falla de energía externa puede desconfigurar el clúster, cuando se apaguen las computadoras y funcione al 50% debido al fallo de corriente. La manera de distribuir las peticiones es eficiente entre las otras máquinas del clúster. [6]

### 2.1.6 Infraestructura de comercio electrónico en alta disponibilidad.

Internet, una red de datos que une al mundo, proporciona gran diversidad de servicios a sus usuarios. Internet abre nuevas oportunidad, y crean nuevos retos, los negocios pueden llegar a un número mayor de clientes, y mantener servicio a las mismas 24 horas al día 365 días al año, en forma continua, alrededor del globo terrestre. Todo esto apoya por tecnología de punta y en la cual se requieren altos niveles de disponibilidad. Una solución de comercio electrónica, tiene tres características claves desde el punto de vista de la infraestructura, que son: alta disponibilidad, escalabilidad, y seguridad. Estas características deben observarse en toda la arquitectura de la solución en cada una de sus capas de hardware y software.

La alta disponibilidad es la habilidad para proveer acceso continuo a los servicios de comercio electrónico para los clientes. Un requisito fundamental para los negocios en Internet disponible 24 horas al día. Examinaremos las opciones que ofrece el mercado para alta disponibilidad a nivel de redes de datos, servidores, sistemas operativos, y bases de datos, también se examina qué se ofrece en las herramientas de administración de redes fundamental para completar la arquitectura de alta disponibilidad. Por último, se tocarán los procedimientos necesarios para asegurar la alta disponibilidad, basados en las mejores prácticas y recomendaciones de los proveedores líderes en el mercado. Lo mejor en hardware y software no podrán asegurar la alta disponibilidad de un sitio de comercio electrónico, si no son correctamente administrados.

El objetivo general de la presente investigación es Dar a conocer como construir una infraestructura de alta disponibilidad para comercio electrónico utilizando tecnología de punta, los objetivos específicos son describir la arquitectura de la infraestructura de una solución de comercio electrónico, y sus componentes, describir la alta disponibilidad a nivel de los componentes y servicios de red, describir la alta disponibilidad a nivel de los servidores Web, describir la alta disponibilidad a nivel de los servidores de base de datos, describir la alta disponibilidad a nivel del diseño de la aplicación, presentar laboratorios prácticos que demuestren los conceptos, y clarifiquen las ideas, el trabajo se llevará a cabo alrededor de la tecnología vigente al momento de desarrollar la tesis.

Las conclusiones de la presente investigación fueron: La arquitectura de la infraestructura de una solución de comercio electrónico puede dividirse en capas

de servicio. Las cuales consisten en las capas de servicios de red, sistema operativo, bases de datos y aplicaciones. Para conseguir un nivel de alta disponibilidad en cada una de ellas, hay que trabajar en la redundancia de componentes en menor o mayor medida, considerando la variable costo/beneficio. En los servicios de red la alta disponibilidad se coloca alrededor de sus componentes, entre los que se encuentran balanceadores de carga geográficos, ruteadores, caché de contenidos, switch, firewalls, y balanceadores de carga a nivel de servidores. La alta disponibilidad se logra al colocar redundancia en los elementos mencionados.

Como parte de los componentes vitales dentro de una solución de comercio electrónico se encuentra los servidores, computadoras especializadas que cumplen una función en la arquitectura de comercio electrónica. El avance tecnológico ha permitido que la arquitectura de los servidores cada día brinde nuevas opciones de alta disponibilidad a un menor costo, haciéndolo accesible a una gran cantidad de usuarios. El clúster de servidores ha sido una tecnología de alta disponibilidad que ha existido por varios años en el mercado. Esta ha alcanzado madurez que ha permitido que se coloquen este tipo de soluciones en ambiente de misión crítica. Existiendo diversas opciones en el mercado de fabricantes como Microsoft®, Novell®, Linux®, y varias de ambientes Unix. Las bases de datos son un componente muy importante dentro de la arquitectura de comercio electrónico y la disponibilidad es crítica para la solución. Dentro de las opciones que existen y se utilizan ampliamente en el mercado está el clúster de Microsoft SQL server, y real application clúster de Oracle. Para garantizar la disponibilidad del sitio de comercio electrónico es necesario un conjunto de herramientas de administración que permitan tomar acciones correctivas y preventivas para mantener la solución trabajando en forma continua. La solución de alta disponibilidad no se puede completar si no se cuenta con un conjunto de políticas y procedimientos que garantice que los planes y diseños propuestos operen de acuerdo con sus especificaciones.[7]

### 2.1.7 Arquitecturas para la alta disponibilidad de cortafuegos con estados.

Los servicios de red se despliegan normalmente sobre plataformas off-the-shelf que suelen carecen de mecanismos de detección de fallos. Es por ello que son susceptibles de sufrir periodos de indisponibilidad debido a fallos en el sistema. Las soluciones clásicas propuestas en las ultimas décadas proponen el uso de mecanismos de redundancia física y monitorización para implementar alta

disponibilidad. Concretamente, la idea consiste en desplegar el mismo servicio sobre distintas réplicas y un protocolo de consenso garantiza que, en cualquier instante de tiempo, al menos una de ellas está procesando las peticiones provenientes de los clientes. Si por cualquier razón la réplica activa sufre un fallo, una nueva replica que estaba actuando como respaldo se selecciona para continuar proporcionando el servicio.

La redundancia física no es suficiente para garantizar la disponibilidad de un servicio implementado por un software con estados puesto que la réplica desconoce los estados de las peticiones que estaba manejando la réplica que ha sufrido un fallo. Es por ello que las réplicas de respaldo no pueden recuperar el servicio apropiadamente puesto que desconocen el estado de las peticiones circulantes. Por esta razón, se debe garantizar que la probabilidad de que los estados sobrevivan a fallos sea alta con el fin de ofrecer una alta disponibilidad. Durante décadas, las primitivas atómicas de broadcast y los esquemas de replicación estudiados en bases de datos aseguran que un conjunto de réplicas se mantendrán consistentes siempre que apliquen las mismas transacciones en el mismo orden[8]. Sin embargo, tales esquemas basados en transacciones síncronas introducen un retardo considerable en las respuestas a los clientes, un precio a pagar para garantizar que todas las réplicas son copias equivalentes. Esto hace que dichos esquemas no sean aplicables a otros escenarios en los que la principal preocupación sea ofrecer un alto rendimiento.

En este trabajo proponemos una arquitectura y una serie de mejoras en entornos en los que el rendimiento y la disponibilidad sean la principal preocupación: Se deben garantizar respuestas rápidas a los clientes a la vez que agilidad en la recuperación de fallos. La naturaleza de estos escenarios hace particularmente difícil la definición e implementación de una solución completa. Algunos ejemplos de software de red con estados presentes en nuestra vida diaria en los que el alto rendimiento sea la principal preocupación son: Nueva generación de firewalls y routers, los cuales definen una política de filtrado basada en el estado de la conexión frente a las aproximaciones estáticas de la primera generación. Redes privadas virtuales, en inglés Virtual Private Network (VPN), que ofrecen túneles seguros a través de internet para comunicar dos redes distantes que pertenecen a la misma organización. Telefonía por internet, también conocido por VoIP, que permite a dos extremos realizar llamadas telefónicas a través de internet. Redes activas, que permiten a sus usuarios inyectar programas adaptados que

despliegan algún tipo de servicio sobre el tráfico que circula por la red. Aunque este trabajo se centra particularmente en los firewalls con estados como caso concreto de estudio, hemos observado que la semántica de los servicios anteriormente enumerados es similar, por tanto, las contribuciones descritas en este trabajo son igualmente aplicables a éstos. El objetivo general de este trabajo es el de proponer mejoras para aumentar la disponibilidad del software con estados en entornos en los que el alto rendimiento es un requisito a satisfacer. Ofrecer una solución completa de alta disponibilidad en estos entornos se convierte en una tarea particularmente difícil puesto que es necesario evaluar las técnicas propuestas con el fin de asegurar que el servicio no reduce el rendimiento ofrecido.

Las conclusiones de la presente investigación son las siguientes: En este trabajo hemos propuesto una arquitectura completa para software de red de alto rendimiento con estados, un serie de protocolos de replicación, así como optimizaciones que reduzcan la cantidad de estados replicados, para de esta forma, disminuir los recursos dedica- dos a la replicación, principalmente el consumo de CPU. La solución propuesta mantiene en mente rendimiento y disponibilidad como principales preocupaciones: se garantiza que las respuestas a clientes son rápidas y los tiempos de recuperación desde fallo son bajos.

Las optimizaciones están extraídas de la semántica del software con estados descrito en el modelo formalizado. La primera de ellas se basa en la observación de que los estados finales Sn no mejoran la disponibilidad, por tanto su replicación puede ser pospuesta indefinidamente. La segunda consiste en el uso de técnicas de aprendizaje supervisado con el fin de reducir el número de estados replicados, garantizando que aquellos estados en los que una petición permanece prolongadamente son estados candidatos a ser replicados. Descartando la replicación de aquellos estados que no aumentan la durabilidad de manera sustancial.

Actualmente estamos trabajando para evaluar las optimizaciones propuestas, así como para mejorar la arquitectura y los protocolos de replicación detallados en este trabajo. Queda pen- diente el estudio a fondo de una solución de replicación basada en la detección y diagnosis de errores que permita la migración de los estados cuando se produzca un fallo, evitando de esta forma la penalización inherentemente asociada a la replicación. De cara a la evaluación, planeamos completar la evaluación de todos los protocolos propuestos así como implementar

escenarios avanzados de balanceo y compartición de carga entre las diferentes replicas.[9]

### 2.1.8 Implementación de un Clúster de Alta Disponibilidad de Base de Datos para GEDGAPA.

La realización de este proyecto permitió implementar un nuevo tipo de tecnología, la tecnología de clúster, dicha tecnología ofrece un sistema de alta disponibilidad para las bases de datos que interactúan con la geDGAPA; misma de la que se carecía y producía tiempos fuera de servicio de las aplicaciones que forzosamente necesitaban establecer conexión con las bases de datos. Ahora el clúster satisface ampliamente las necesidades que se presentan en el manejo diario de los servicios que brinda la geDGAPA al personal académico de la Universidad.

Todos los componentes de software del clúster son de Licencia Pública General, lo que significa que es software libre de código abierto, por lo tanto, la implementación del clúster tiene un mérito sobresaliente, porque hoy en día pocas son las instituciones que toman el riesgo de ir de la mano con el software libre y probar que funciona adecuadamente, ahorrando recursos en la compra de licencias de software comercial y soporte técnico; por ello , la DGAPA es una institución que va más allá, apoyando e impulsando nuevas tecnologías que fomenten el progreso de sí y de los servicios que brinda.

Para implementar el clúster era necesario contar con recursos de hardware y software más eficientes de los que se tenían; por dicho motivo fue necesario migrar el servidor de bases de datos a un servidor más potente y con mayor capacidad, un Sun Fire X4200; con este servidor se obtiene un mejor rendimiento, porque tiene más memoria RAM y un procesador de doble núcleo que ofrece más velocidad de respuesta a los procesos.

El clúster proporciona redundancia tanto en hardware como en software; en hardware porque se tienen tres servidores que se comunican y funcionan como uno solo de manera transparente para los usuarios. En software, los mismos servidores pueden actuar como replicador cuando sea necesario, es decir, cuando el nodo maestro no funcione adecuadamente y se realice un proceso de failover, cualquiera de los otros dos nodos es capaz de reemplazar al nodo maestro,

porque cuenta con las herramientas y configuraciones necesarias para poder satisfacer las peticiones de consulta a la base de datos por parte de los clientes.

La planeación que se le dio a la realización de este proyecto permitió lograr satisfactoriamente la puesta en marcha del clúster. Cada una de las etapas del proyecto, fueron fundamentales para que todo el trabajo en conjunto resultará en el éxito de los objetivos planteados para resolver la problemática inicial, acerca de la sobrecarga del servidor de bases de datos, que mermaba el funcionamiento de las aplicaciones, al tener que responder a todas las peticiones de acceso a las bases de datos en un tiempo inadecuado.
Las ventajas obtenidas de la realización de este proyecto son:

- Se tiene un sistema de replicación de bases de datos que proporciona alta disponibilidad de las mismas.

- Se tienen medidas preventivas, como el proceso de failover, que actúa ante la presencia de una falla en el nodo maestro.

- Se cuenta con una nueva versión de DBMS de PostgreSQL que permite mantener a la geDGAPA con un software actual.

- Con software de Licencia Pública General se brinda a la geDGAPA de una alta tecnología.

- Se implementaron estrategias nuevas de monitoreo que facilitan la prevención e identificación de fallas.

- El rendimiento de los sistemas que acceden a la base de datos es más ágil; porque ahora se tiene tres nodos que pueden responder a peticiones de lectura de los datos.

Con base en las pruebas realizadas, se concluye que el clúster es capaz de brindar la replicación de la base de datos dbgdgapa de manera exitosa, proporcionando redundancia de la misma en dos nodos extras disponibles para su uso cuando se necesite.

Los resultados fueron satisfactorios a medida de las necesidades a resolver de la dependencia, sin embargo, también hay que tener presente los componentes del clúster presentan debilidades y cuales otros pueden mejorarse.[10]

## 2.2 Bases Teóricas.

### 2.2.1 Clúster

#### a) Definición.

El término clúster se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de componentes de hardware comunes y que se comportan como si fuesen una única computadora. Hoy en día desempeñan un papel importante en la solución de problemas de las ciencias, las ingenierías y del comercio moderno.

La tecnología de clúster 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.

El cómputo con clúster 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. Simplemente, un clúster es un grupo de múltiples ordenadores unidos mediante una red de alta velocidad, de tal forma que el conjunto es visto como un único ordenador, más potente que los comunes de escritorio. Los clúster son usualmente empleados para mejorar el rendimiento y/o la disponibilidad por encima de la que es provista por un solo computador típicamente siendo más económico que computadores individuales de rapidez y disponibilidad comparables.[11]

El término clúster (del inglés clúster, "grupo" o "racimo") se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de hardwares comunes y que se comportan como si fuesen una única computadora. La tecnología de clústeres 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. El cómputo con clústeres 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. Simplemente, un clúster es un grupo de múltiples ordenadores unidos mediante una red de alta velocidad, de tal forma que el conjunto es visto como un único ordenador, más potente que los comunes de escritorio. Los clústeres son usualmente empleados para mejorar el rendimiento y/o la disponibilidad por encima de la que es provista por un solo computador típicamente siendo más económico que computadores individuales de rapidez y disponibilidad comparables.[12]

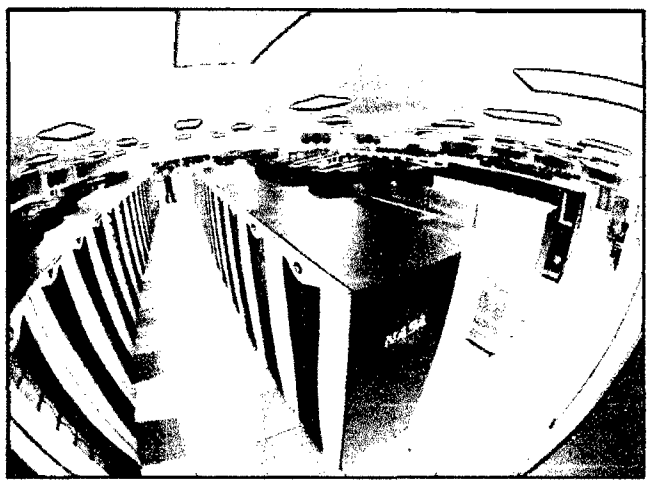


Figura 1. Clúster de Servidores de la NASA. [12]

## b) Componentes de un Clúster.

En general, un clúster necesita de varios componentes de software y hardware para poder funcionar:

- Nodos.
- Almacenamiento.
- Sistemas operativos.
- Conexiones de red.

- ❖ Middleware.
- ❖ Protocolos de comunicación y servicios.
- ❖ Aplicaciones.
- ❖ Ambientes de programación paralela.

**b.1. Nodos.**

Los nodos tienen las siguientes características:

- ✓ Todo nodo tiene acceso a todos los datos de la configuración de clúster.
- ✓ Todo nodo se comunica con los demás nodos del clúster a través de una o más redes físicamente independientes. Los adaptadores de red, conocidos como *interfaces de red* en los clústeres de servidor, conectan los nodos a las redes.
- ✓ Todos los nodos del clúster saben cuándo otro sistema se une o abandona el clúster.
- ✓ Todos los nodos del clúster tienen conocimiento de los recursos que se ejecutan localmente y de los recursos que se ejecutan en los demás nodos del clúster.
- ✓ Todos los nodos del clúster están agrupados bajo un nombre común, el nombre de clúster, que se utiliza para acceder al clúster y para gestionarlo.

Cuando se inicia un nodo, éste busca nodos activos en las redes designadas para la comunicación interna. Si encuentra un nodo activo, intenta unirse al clúster del nodo. Si no encuentra ningún clúster, intenta formar un clúster tomando control del recurso de quórum. El recurso de quórum almacena la versión más reciente de la base de datos del clúster, que contiene la configuración del clúster y los datos de estado. Un clúster de servidor mantiene una copia coherente y actualizada de la base de datos del clúster en todos los nodos activos. Un nodo puede albergar unidades físicas o lógicas, denominadas recursos. Los administradores organizan estos recursos de clúster en unidades funcionales conocidas como grupos y asignan estos grupos a nodos individuales. Si un nodo falla, el clúster de servidor transfiere

los grupos alojados en el nodo a otros nodos del clúster. Este proceso de transferencia se llama sustitución por anomalía.

El proceso inverso, recuperación tras error, tiene lugar cuando el nodo con errores se vuelve a activar y los grupos que habían sido sustituidos por anomalía se transfieren de nuevo al nodo original.[13]

**b.2. Almacenamiento.**

El almacenamiento puede consistir en una NAS, una SAN, o almacenamiento interno en el servidor. El protocolo más comúnmente utilizado es NFS (Network File System), sistema de ficheros compartido entre servidor y los nodos. Sin embargo existen sistemas de ficheros específicos para clústeres como Lustre (CFS) y PVFS2.

Tecnologías en el soporte del almacenamiento en discos duros:

- ✓ IDE o ATA: velocidades de 33, 66, 100, 133 y 166 MB/s
- ✓ SATA: velocidades de 150, 300 y 600 MB/s
- ✓ SCSI: velocidades de 160, 320, 640 MB/s. Proporciona altos rendimientos.
- ✓ SAS: aúna SATA-II y SCSI. Velocidades de 300 y 600 MB/s
- ✓ Las unidades de cinta (DLT) son utilizadas para copias de seguridad por su bajo costo.

NAS (Network Attached Storage) es un dispositivo específico dedicado al almacenamiento a través de red (normalmente TCP/IP) que hace uso de un sistema operativo optimizado para dar acceso a través de protocolos CIFS,NFS, FTP o TFTP.

Por su parte, DAS (Direct Attached Storage) consiste en conectar unidades externas de almacenamiento SCSI o a una SAN (storage area network: red de área de almacenamiento) a través de un canal de fibra. Estas conexiones son dedicadas. Mientras NAS permite compartir el almacenamiento, utilizar la red, y tiene una gestión más sencilla, DAS proporciona mayor rendimiento y mayor fiabilidad al no compartir el recurso.[12]

### b.3. Sistemas operativos.

Un sistema operativo (OS del inglés Operating System) es un programa o conjunto de programas que en un sistema informático gestiona los recursos dehardware y provee servicios a los programas de aplicación, ejecutándose en modo privilegiado respecto de los restantes y anteriores próximos y viceversa (aunque puede que parte del mismo se ejecute en espacio de usuario).

Uno de los propósitos del sistema operativo que gestiona el núcleo intermediario consiste en gestionar los recursos de localización y protección de acceso del hardware, hecho que alivia a los programadores de aplicaciones de tener que tratar con estos detalles. La mayoría de aparatos electrónicos que utilizan microprocesadores para funcionar, llevan incorporado un sistema operativo (teléfonos móviles, reproductores de DVD, computadoras, radios, enrutadores, etc.). En cuyo caso, son manejados mediante una interfaz gráfica de usuario, un gestor de ventanas o un entorno de escritorio, si es un celular, mediante una consola o control remoto si es un DVD y, mediante una línea de comandos o navegador web si es un enrutador.[14]

### b.4. Conexiones de red.

Los nodos de un clúster pueden conectarse mediante una simple red Ethernet con placas comunes (adaptadores de red o NICs), o utilizarse tecnologías especiales de alta velocidad como Fast Ethernet, Gigabit Ethernet, Myrinet, InfiniBand, SCI, etc.

***Ethernet.*** Son las redes más utilizadas en la actualidad, debido a su relativo bajo coste. No obstante, su tecnología limita el tamaño de paquete, realizan excesivas comprobaciones de error y sus protocolos no son eficientes, y sus velocidades de transmisión pueden limitar el rendimiento de los clústeres. Para aplicaciones con paralelismo de grano grueso puede suponer una solución acertada. La opción más utilizada en la actualidad es Gigabit Ethernet (1 Gbit/s), siendo emergente la solución 10 Gigabit Ethernet (10 Gbit/s). La latencia de estas tecnologías está en torno a los 30 a 100 µs, dependiendo del protocolo de comunicación empleado. En todo caso, es la red de administración por excelencia, así que aunque no sea la solución de red de

altas prestaciones para las comunicaciones, es la red dedicada a las tareas administrativas.

***Myrinet (Myrinet 2000 y Myri-10G).*** Su latencia es de 99 a 10 µs, y su ancho de banda es de 2 a 10 Gbit/s (para Myrinet 2000 y Myri-10G, respectivamente). Es la red de baja latencia más utilizada en la actualidad, tanto en clústeres como en MPP; está presente en más de la mitad de los sistemas del top500. Tiene dos bibliotecas de comunicación a bajo nivel (GM y MX). Sobre estas bibliotecas están implementadas MPICH-GM, MPICH-MX, Sockets-GM y Sockets MX, para aprovechar las excelentes características de Myrinet. Existen también emulaciones IP sobre TCP/IP, IPoGM e IPoMX.

***InfiniBand.*** Es una red surgida de un estándar desarrollado específicamente para realizar la comunicación en clústers. Una de sus mayores ventajas es que mediante la agregación de canales (x1, x4 y x12) permite obtener anchos de banda muy elevados. La conexión básica es de 2 Gbit/s efectivos y con 'quad connection' x12 alcanza los 96 Gbit/s. No obstante, los startups no son muy altos, se sitúan en torno a los 10 µs. Define una conexión entre un nodo de computación y un nodo de I/O. La conexión va desde un Host Channel Adapter (HCA) hasta un Target Channel Adapter (TCA). Se está usando principalmente para acceder a arrays de discos SAS.

***SCI (scalable coherent interface) IEEE standard 1596-1992.*** Su latencia teórica es de 1,43 µs y su ancho de banda de 5333 Mbit/s bidireccional. Al poder configurarse con topologías de anillo (1D), toro (2D) e hipercubo (3D) sin necesidad de switch, se tiene una red adecuada para clústers de pequeño y mediano tamaño. Al ser una red de extremadamente baja latencia, presenta ventajas frente a Myrinet en clústeres de pequeño tamaño al tener una topología punto a punto y no ser necesaria la adquisición de un conmutador. El software sobre SCI está menos desarrollado que sobre Myrinet, pero los rendimientos obtenidos son superiores, destacando SCI Sockets (que obtiene startups de 3 microsegundos) y ScaMPI, una biblioteca MPI de elevadas prestaciones. Además, a través del mecanismo de preloading (LD_PRELOAD) se puede conseguir que todas las comunicaciones del sistema vayan a través de SCI-SOCKETS (transparencia para el usuario).[12]

## b.5. Middleware.

El middleware se define como un software distribuido constituido por un conjunto de recursos y servicios que permiten a múltiples procesos, corriendo en una o más máquinas, interactuar a través de una red. Es una colección no estructurada de recursos y servicios, ubicados entre la capa de transporte y la capa de aplicación, que pueden ser utilizados individualmente o conjuntamente por los procesos de la aplicación. Provee una interface abstracta que ofrece al programador de aplicaciones una visión uniforme de elementos heterogéneos de bajo nivel, como sistemas operativos y redes subyacentes.

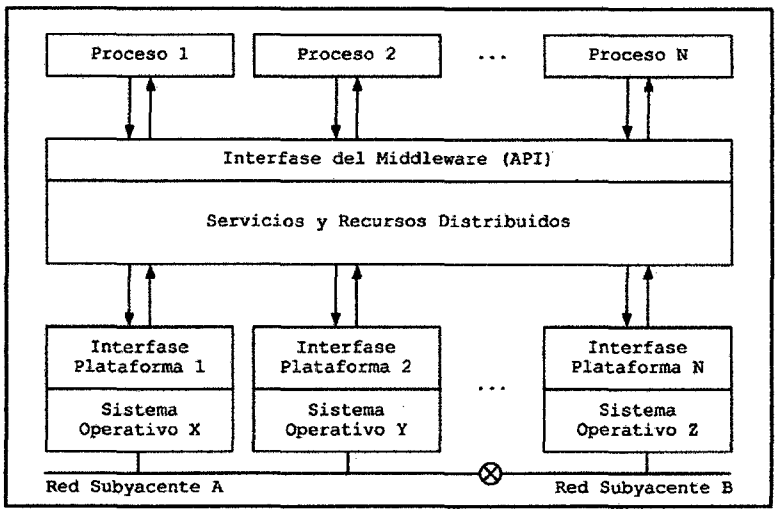



Figura 2. Middleware.[15]

El middleware debe presentar las siguientes características:

Interoperabilidad: es la habilidad de dos o más sistemas o componentes de intercambiar información y utilizarla. El middleware debe permitir el intercambio de información entre hardware y sistemas operativos distintos e inclusive, entre implementaciones de distintos fabricantes (siempre que adhieran a un mismo estándar).

Transparencia: se deben ocultar las diferencias de las capas subyacentes de forma tal que el middleware sea visto en forma consistente por las aplicaciones y procesos que se construyen sobre él. Por ejemplo, debe ser posible ubicar servicios y recursos distribuidos sin recurrir a direcciones de red.

Confiabilidad y Disponibilidad: un middleware es confiable si es capaz de cumplir con la funcionalidad que se le requiere bajo ciertas condiciones durante un período específico de tiempo. La disponibilidad es la capacidad de prestar un servicio correcto. Para lograr middlewares confiables y de alta disponibilidad, es conveniente la distribución de recursos y servicios evitando componentes centralizados. No puede suceder que por fallar una máquina o un proceso, todo el middleware quede inutilizado.

Escalabilidad: es una medida de la facilidad con la que un sistema o componente puede ser modificado para acomodarse a la envergadura del problema a resolver. A medida que crece la cantidad de procesos de aplicación, el middleware no debe perder funcionalidad ni performance en forma excesiva.

Abstracción: un middleware tiene que facilitar la construcción de aplicaciones distribuidas. Debe proveer un mayor nivel de abstracción que las capas inferiores de forma que el desarrollador de aplicaciones, en lugar de manejar entidades como conexiones, puertos o semáforos, trabaje con abstracciones de mayor nivel como pueden ser objetos, eventos y transacciones.[15]

### b.6. Ambientes de programación paralela.

Los ambientes de programación paralela permiten implementar algoritmos que hagan uso de recursos compartidos: CPU (central processing unit: unidad central de proceso), memoria, datos y servicios.

## c) Clasificación de los Clústeres.[16-18]

Los clústers dependiendo de su aplicabilidad pueden clasificarse de diferentes maneras. La clasificación más generalizada es la que se presenta a continuación:

- Alto rendimiento (HP, high performance).
- Alta disponibilidad (HA, high availability).
- Balanceo de Carga (Load Balancing).
- Alta Confiabilidad (HR, high reliability).

### c.1. Alto rendimiento (HP, high performance).

Este tipo de clúster lo que busca es suplir las necesidades de súper computación para resolver problemas de determinadas aplicaciones que requieren un alto procesamiento, esto se logra mediante la utilización de un grupo de máquinas individuales las cuales son interconectadas entre sí a través de redes de alta velocidad y de esta manera se obtiene un sistema de gran rendimiento que actúa como uno solo.

La utilidad principal de este tipo de *clúster* es principalmente en aplicaciones en las que se requieren gran capacidad de procesamiento computacional, la cual soluciona problemas de alto procesamiento mediante la utilización de técnicas necesarias para la paralelización de la aplicación, distribución de los datos a los nodos, la obtención y presentación de resultados finales.

Generalmente estos problemas de cómputo suelen estar ligados a:

- Cálculos matemáticos.
- Estado del tiempo.
- Cifrado y Descifrado de códigos.
- Compresión de datos.
- Astronomía.
- Simulación Militar.

### c.2. Alta disponibilidad (HA, high availability).

Clúster muy solicitado y de mucha importancia para empresas que brindan servicios 24-7 dónde su principal función es la de mejorar los servicios que dichas empresas ofrecen a los clientes en las redes a las que pertenecen, sean estas internas (intranet) o externas (Internet).

El brindar alta disponibilidad no hace referencia a conseguir una gran capacidad de cálculo, si no lograr que una colección de máquinas funcionen en conjunto y que todas realicen la misma función que se les encomendó. La característica principal de este clúster es que ante la existencia de algún problema o fallo de uno de los nodos, el resto asumen ese fallo y con ello las

tareas del nodo con problemas. Estos mecanismos de alta disponibilidad lo brindan de forma transparente y rápida para el usuario.

La escalabilidad en un clúster de alta disponibilidad se traduce en redundancia lo cual garantiza una pronta recuperación ante cualquier fallo.
La flexibilidad y robustez que poseen este tipo de clúster los hacen necesario en sistemas cuya funcionalidad principal es el intercambio masivo de información y el almacenamiento de datos sensibles, dónde se requiere que el servicio esté presente sin interrupciones. El mantenimiento es otra de las ventajas que ofrece. El mantenimiento se puede realizar de manera individual a cada máquina que compone el conglomerado evitando comprometer los servicios que este brinda.

Existen dos tipos de configuraciones aplicables a estos clústeres:

- ***Configuración activo_pasivo:*** Esta configuración tiene dos actividades en los nodos que componen el clúster, los activos son aquellos que se encargan de ejecutar las aplicaciones encomendadas, mientras que los nodos restantes actúan como respaldos redundantes para los servicios ofrecidos.

- ***Configuración activo - activo:*** En este caso, todos los nodos actúan como servidores activos de una o más aplicaciones y potencialmente como respaldos para las aplicaciones que se ejecutan en otros nodos. Cuando un nodo falla las aplicaciones que se ejecutaba en él migran a uno de los nodos de respaldo.

**c.3. Balanceo de Carga (Load Balancing).**

Técnica muy utilizada para lograr que un conjunto de servidores de red compartan la carga de trabajo y con ello el tráfico de sus clientes. Este proceso de dividir la carga de trabajo entre los servidores reales permite obtener un mejor tiempo de acceso a las aplicaciones y con ellos tener una mejor confiabilidad del sistema. Además como es un conjunto de servidores el que atiende el trabajo, la falla de uno de ellos no ocasiona una falla total del sistema ya que las funciones de uno, las puede suplir el resto.

El balanceo de carga permite suplir las necesidades del inminente crecimiento del tráfico en Internet. Existen dos alternativas para manipular el crecimiento del tráfico de Internet: la primera permite usar una máquina de grandes características y de alto precio que probablemente a futuro quede obsoleta; la segunda alternativa consiste en utilizar un conjunto de servidores virtuales o granja de servidores de bajo costo que trabajan en conjunto para balancear la carga entre ellos.

El balanceo de carga consiste en compartir la carga de trabajo y tráfico de los clientes que éstos acceden. Al grupo de servidores que prestan este servicio se los conoce como servidores virtuales. Al balancear la carga se mejora el tiempo de respuesta, acceso y confiabilidad. La caída de un servidor no influye en el funcionamiento de todo el *clúster* ya que las funciones de éste son asumidas por el resto de servidores virtuales. Cuando la carga de trabajo sea mayor se pueden añadir más servidores al *clúster* y escalar este sistema para garantizar el balanceo de carga. Existen diferentes métodos de distribución de carga que se detallan a continuación.

**c.4. Alta Confiabilidad (HR, high reliability).**

Clúster caracterizado por ofrecer una alta confiabilidad al sistema. La idea es obtener respuestas eficientes del sistema a pesar de tener una sobrecarga de las capacidades de un servidor. Estos clústeres se caracterizan por ejecutar un mayor número de tareas en el menor tiempo posible.

**d) Funcionamiento.[19]**

El funcionamiento de un *clúster* está basado en dos partes fundamentales: a nivel de software y a nivel de hardware.

***A nivel de Software.*** Mediante el uso de un sistema operativo que brinde las herramientas necesarias para la creación de un clúster , tal es el caso de un kernel Linux modificado, compiladores y aplicaciones especiales, los cuales permitan que los programas que se ejecutan en el sistema exploten todas las ventajas del clúster.

***A nivel de Hardware.*** Mediante la interconexión entre máquinas (nodos) del clúster, las cuales se juntan utilizando redes dedicadas de alta velocidad como por ejemplo Gigabit Ethernet.

Cuando se trata de brindar balanceo de carga mediante un clúster el hardware y software trabajan conjuntamente para distribuir la carga de tráfico a los nodos, para de esta manera poder atender eficientemente las subtareas encomendadas y con ello la tarea general asignada al clúster. Un servicio de alta disponibilidad en el clúster normalmente no distribuye la carga de tráfico a los nodos (balanceo de carga) ni comparte la carga de procesamiento (alto rendimiento) sino más bien su función es la de estar preparado para entrar inmediatamente en funcionamiento en caso de que falle algún otro servidor.

## d.1. Características y funcionamiento.

### d.1.1. *Sistemas de alta disponibilidad y sistemas tolerantes a fallos.*

Los sistemas tolerantes a fallos son aquellos sistemas que solucionan problemas de fallo del hardware de algún dispositivo, la tolerancia a fallos hace que un sistema pueda detectar y actuar instantáneamente para restablecer el servicio brindado. Los sistemas de alta disponibilidad como ya se vio en el apartado anterior, involucra el tener servidores que actúan entre ellos como respaldos vivos de la información que sirven. Este tipo de clústeres se les conoce también como clúster de redundancia. En los últimos años la idea de alta disponibilidad y de tolerancia a fallos se ha ido acercando, con el surgimiento de nuevas tecnologías y el abaratamiento del hardware se ha logrado que con la idea de alta disponibilidad se logre tener un sistema tolerante a fallos a bajo precio.

### *d.1.2. SPOF ( Single Point of Failure ó Punto Simple de Fallo).*

SPOF hace referencia a la tenencia de un elemento no replicado que puede estar sujeto a fallos, logrando con esto la suspensión del servicio que se está brindando. Es por ello la importancia de evitar tener un SPOF en los subsistemas del sistema general, ya que con ello se pondría en peligro la prestación continua de servicios del sistema. En sistemas de alta disponibilidad a mas de tener redundancia en sus servidores, es importante tenerla en otros dispositivos que componen el clúster, tal es el caso de

dispositivos de interconexión, red de comunicación de servidores; etc. Esto con la finalidad de evitar el tener un SPOF a nivel de subsistemas y sistemas como tal que conforman el clúster que se está implementando. El servicio de datos hace referencia al servicio y sus respectivos recursos que se esté brindando a un cliente. En entornos de alta disponibilidad al servicio brindado y al grupo de recursos se denominan logical host o software package. Los recursos que se estén utilizando deben tener mecanismos necesarios que permitan la suplantación y conmutación física entre los nodos cuando uno de estos falle, logrando de esta manera que el servicio de datos ofrecido falle en su funcionamiento. De esta manera la única afección que el sistema tendrá es en el tiempo de conmutación en la puesta en marcha del servicio de datos.

### d.1.3.*Dinámica de Alta Disponibilidad.*

Dinámica que hace referencia a las reconfiguraciones que el clúster debe hacer para garantizar la máxima disponibilidad de un determinado sistema; va orientada a los nodos que conforman el clúster y la forma de cómo éste responde. Existen diferentes maneras de cómo el sistema responde ante la presencia de un fallo, entre las cuales se tiene:

***Tolerancia a fallos (failover).*** Se da cuando un nodo falla y otro debe asumir sus responsabilidades. Para ello el nodo entrante debe importar los recursos del nodo con fallo y habilitar los servicios de datos.

***Toma de control o tolerancia a fallos automático (takeover).*** Se produce cuando el servicio de datos falla y se detecta por un determinado nodo, a este nodo se lo considera nodo fallido y se ve forzado a ceder sus servicios y recursos. En este caso se requiere una monitorización continua del servicio de datos.

***Tolerancia a fallos manual (switchover o giveaway).*** Se caracteriza por ceder los recursos y servicios de datos de un nodo a otro. Se utiliza cuando se realizan tareas de mantenimiento y administración a un nodo.

***División de cerebros (splitbrain).*** Mecanismo utilizado cuando el proceso de gestión de un clúster HA(High Availability) falla. Esta falla se da debido a

problemas en la comunicación y verificación de los nodos existentes. Debido a que los nodos no conocen de la existencia de sus vecinos y asumen que son los únicos en el sistema, por tal motivo cada nodo intentará apropiarse de todos los recursos del sistema incluyendo el servicio de datos y tener el control total del cluster. El *splitbrain* es un caso especial del *fileover*.

El splitbrain puede dejar a un sistema fuera de funcionamiento. Se puede evitar utilizando dos métodos. El primero es actuar de forma prudente ante esta falla, utilizando los recursos compartidos como señal de estar activos, luego de que un nodo constata el problema debe reservar un recurso compartido llamado quórum, este recurso debe ser reservado por un solo nodo y el nodo que llegue tarde a la reserva entiende que debe abandonar el clúster y ceder todos sus recursos. El recurso quórum únicamente es utilizado como método de decisión para abandonar o no un clúster. El segundo método consiste en tratar de dejar fuera o apagar al nodo con fallo una vez detectado el problema, el primer nodo que lo haga toma el control del clúster y con ello el control de todos los recursos.

***Recursos de un Servicio de Datos.*** Al trabajar con un clúster de alta disponibilidad se tienen diferentes tipos de recursos que son fundamentales para su funcionamiento y que se listan a continuación:

- Recursos Computacionales. Son aquellos que permiten alojar y ejecutar el servicio de datos brindado. Estos recursos se consideran a nivel de CPU y el clúster. En alta disponibilidad se tienen recursos a nivel de clúster dónde cada nodo que lo conforma debe tener una copia en memoria del programa de servicio de datos.

- Recursos de Comunicaciones. Recursos utilizados para brindar el acceso al servicio de datos mediante una red de comunicaciones.

- Recursos de Almacenamiento. Son los recursos más críticos en alta disponibilidad. Se debe garantizar la integridad y confiabilidad de los datos almacenados en los discos de los nodos o servidores. La falla de estos dispositivos hace que los datos se corrompan y con ello se tenga efectos irreversibles que afecten el rendimiento del sistema.

### 2.2.2 Alta Disponibilidad de la Información.

#### a) Definición.[20, 21]

Para un servicio o aplicación de software específico, la alta disponibilidad se mide en última instancia en función de las expectativas y la experiencia del usuario final. El impacto en el negocio tangible y percibido del tiempo de inactividad se puede expresar en términos de pérdida de información, daños a la propiedad, disminución de la productividad, costos de oportunidad, daños contractuales o pérdida de la buena fe.

El principal objetivo de una solución de alta disponibilidad es minimizar o mitigar el impacto del tiempo de inactividad. Una estrategia eficaz con este fin equilibra de manera óptima los procesos empresariales y los contratos de nivel de servicio (SLA) con las capacidades técnicas y los costos de infraestructura.

Se considera que una plataforma tiene una alta disponibilidad según el contrato, y las expectativas de los clientes y las partes interesadas. La disponibilidad de un sistema se puede expresar mediante este cálculo:

$$Tiempo\ activo\ real/tiempo\ activo\ esperando\ x\ 100\%$$

| Porcentaje de disponibilidad | Tiempo de inactividad anual total | Cantidad de nueves |
|---|---|---|
| 99% | 3 días, 15 horas | 2 |
| 99,9% | 8 horas, 45 minutos | 3 |
| 99,99% | 52 minutos, 34 segundos | 4 |
| 99, 999% | 5 minutos, 15 segundos | 5 |

Tabla 1. Porcentajes y tiempos para alta disponibilidad.

El valor resultante se suele expresar en el sector en función de la cantidad de nueves que ofrece la solución. Se pretende expresar un número anual de minutos de tiempo activo posible o, por el contrario, de minutos de tiempo de inactividad.

Alta disponibilidad (High Availability) es un protocolo de diseño del sistema y su implementación asociada que asegura un cierto grado absoluto de continuidad operacional durante un período de medición dado. Disponibilidad se refiere a la habilidad de la comunidad de usuarios para acceder al sistema, someter nuevos trabajos, actualizar o alterar trabajos existentes o recoger los resultados de

trabajos previos. Si un usuario no puede acceder al sistema se dice que está no disponible. El término tiempo de inactividad (downtime) es usado para definir cuándo el sistema no está disponible.

**b) Tiempo de inactividad previsto e imprevisto.**

Las interrupciones del sistema pueden ser previstas o planeadas, o el resultado de un error no planeado. No se tiene que considerar el tiempo de inactividad de manera negativa si este se administra correctamente. Existen dos tipos clave de tiempo de inactividad previsto:

- **Mantenimiento planeado.** Se anuncia previamente y se coordina una ventana de tiempo para las tareas de mantenimiento planeado, como revisiones de software, actualizaciones de hardware, actualizaciones de contraseñas, nuevas indizaciones sin conexión, cargas de datos o el ensayo de los procedimientos de recuperación ante desastres. Los procedimientos operativos intencionales y bien administrados deben minimizar el tiempo de inactividad y evitar la pérdida de datos. Las actividades de mantenimiento planeado pueden considerarse inversiones necesarias para evitar o mitigar otros escenarios de interrupciones no planeadas potencialmente más graves.

- **Interrupción no planeada.** Pueden surgir errores de nivel del sistema, de infraestructura o de procesos que no sean planeados o controlables, o que sean imprevistos, pero que su aparición sea muy poco probable o su impacto se considere aceptable. Una solución de alta disponibilidad eficaz detecta estos tipos de errores, se recupera automáticamente de la interrupción y luego restablece la tolerancia a errores.

**c) Cuantificación del tiempo de inactividad.**

Cuando se genera tiempo de inactividad, ya sea previsto o imprevisto, el principal objetivo de la empresa es reanudar el funcionamiento del sistema y minimizar la pérdida de datos. Cada minuto de tiempo de inactividad conlleva costos directos e indirectos. Con el tiempo de inactividad imprevisto, debe equilibrar el tiempo y el esfuerzo necesarios para determinar por qué se produjo la interrupción, cuál es el estado actual del sistema y qué pasos se deben realizar para recuperarse de la interrupción.

En un momento predeterminado en cualquier interrupción, debe fomentar o tomar la decisión empresarial de dejar de investigar la interrupción o de realizar tareas de mantenimiento, recuperarse de la interrupción mediante la reanudación del funcionamiento del sistema y, si es necesario, restablecer la tolerancia a errores.

### c.1. Cálculos Porcentuales.

Disponibilidad es usualmente expresada como un porcentaje del tiempo de funcionamiento en un año dado. En un año dado, el número de minutos de tiempo de inactividad no planeado es registrado para un sistema, el tiempo de inactividad no planificado agregado es dividido por el número total de minutos en un año (aproximadamente 525.600) produciendo un porcentaje de tiempo de inactividad; el complemento es el porcentaje de tiempo de funcionamiento el cual es lo que denominamos como disponibilidad del sistema. Valores comunes de disponibilidad, típicamente enunciado como número de "nueves" para sistemas altamente disponibles son:

- 99,9% = 43.8 minutos/mes u 8,76 horas/año ("tres nueves").
- 99,99% = 4.38 minutos/mes o 52.6 minutos/año ("cuatro nueves").
- 99,999% = 0.44 minutos/mes o 5.26 minutos/año ("cinco nueves").

Es de hacer notar que tiempo de funcionamiento y disponibilidad no son sinónimos. Un sistema puede estar en funcionamiento y no disponible como en el caso de un fallo de red. Se puede apreciar que estos valores de disponibilidad son visibles mayormente en documentos de ventas o marketing, en lugar de ser una especificación técnica completamente medible y cuantificable.

### c.2. Medida e interpretación.

Claramente como la disponibilidad medida está sujeta a algún grado de interpretación. Un sistema que ha estado en funcionamiento por 365 días en un año no bisiesto quizá ha sido eclipsado por un fallo de red que duro 9 horas durante un periodo de uso pico; la comunidad de usuarios verá el sistema como no disponible, mientras el administrador del sistema reclamara el 100%

de "tiempo de funcionamiento". Sin embargo siguiendo la verdadera definición de disponibilidad, el sistema estará aproximadamente 99.897% disponible (8751 horas de time out de las 8760 horas por año no bisiesto).

También sistemas experimentando problemas de rendimiento son frecuentemente estimados como entera o parcialmente no disponibles por los usuarios mientras administradores quizás tengan una diferente (y probablemente incorrecta, ciertamente en el sentido del negocio) percepción. Similarmente no disponibilidad de funciones de aplicación no seleccionadas quizás pasen inadvertidas para administradores sin embargo podrían ser devastadoras para usuarios una verdadera medida de disponibilidad es integral.

Disponibilidad debe ser medida para ser determinada, idealmente con herramientas de monitorización comprensivas ("instrumentación") que son ellas mismas altamente disponibles. Si hay una falta de instrumentación, sistemas soportando un alto volumen de procesamiento de transacciones a través del día y la noche tales como procesamiento de tarjetas de crédito o conmutadores telefónicos, son frecuentemente e inherentemente mejor monitorizados, al menos por los mismos usuarios, que sistemas que experimentan pausas periódicas en la demanda.

### 2.2.3 Clúster de Alta Disponibilidad.[22]

Un clúster 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. Podemos dividirlo en dos clases:

- **Alta disponibilidad de infraestructura:**

  Si se produce un fallo de hardware en alguna de las máquinas del clúster, el software de alta disponibilidad es capaz de arrancar automáticamente los servicios en cualquiera de las otras máquinas del clúster (failover). Y cuando la máquina que ha fallado se recupera, los servicios son nuevamente migrados a la máquina original (failback). Esta capacidad de recuperación automática de servicios nos garantiza la alta disponibilidad de los servicios

ofrecidos por el clúster, minimizando así la percepción del fallo por parte de los usuarios.

- **Alta disponibilidad de aplicación**:

  Si se produce un fallo del hardware o de las aplicaciones de alguna de las máquinas del clúster, el software de alta disponibilidad es capaz de arrancar automáticamente los servicios que han fallado en cualquiera de las otras máquinas del clúster. Y cuando la máquina que ha fallado se recupera, los servicios son nuevamente migrados a la máquina original. Esta capacidad de recuperación automática de servicios nos garantiza la integridad de la información, ya que no hay pérdida de datos, y además evita molestias a los usuarios, que no tienen por qué notar que se ha producido un problema.

No hay que confundir un clúster de alta disponibilidad con un clúster de alto rendimiento. El segundo es una configuración de equipos diseñado para proporcionar capacidades de cálculo mucho mayores que la que proporcionan los equipos individuales (véanse por ejemplo los sistemas de tipo Clúster Beowulf), mientras que el primer tipo de clúster está diseñado para garantizar el funcionamiento ininterrumpido de ciertas aplicaciones.

**a) Alta Disponibilidad para Linux. [23]**

Actualmente Linux es conocido, como un sistema operativo estable; la problemática se genera cuando el hardware, no es tan fiable como se desearía. En la mayoría de los casos, cuando un sistema falla normalmente es debido a un fallo de hardware o a un fallo humano (debido a un error en la administración del sistema). En los casos en que un fallo hardware provoca graves consecuencias, debido a la naturaleza del servicio (aplicaciones críticas), se implementan sistemas tolerantes a fallos (fault tolerant ó FT); en los cuales, el servicio esta siempre activo. El problema de estos sistemas, es que son extremadamente caros y normalmente no hay presupuesto. Además suelen ser soluciones cerradas, totalmente dependientes de la empresa contratada. Se suele poner un servidor tolerante a fallos, varias interfaces de red, con tomas de alimentación redundantes y climatización especial.

Los sistemas de alta disponibilidad (high availability ó HA), intentan obtener prestaciones cercanas al la tolerancia a fallos, pero a un precio muchísimo más interesante. Esta es una opción, que la ha hecho crecer en importancia dentro del mundo empresarial. La alta disponibilidad está basada en la replicación de elementos, mucho más baratos que un sólo elemento tolerante a fallos. Naturalmente, si hablamos de replicar servidores, hablaremos de un clúster de alta disponibilidad. Sistemas tolerantes a fallos los podemos encontrar en entornos muy críticos, tales como una central nuclear o el sistema de navegación de una aeronave moderna.

### a.1. Sistemas de alta disponibilidad y sistemas tolerantes a fallos.

En un sistema tolerante a fallos, cuando se produce un fallo hardware, el hardware asociado a este tipo de sistema es capaz de detectar el subsistema que falla y obrar en consecuencia para restablecer el servicio en segundos (o incluso décimas de segundo).

El cliente del servicio no notará ningún tiempo de fuera de servicio. En los sistemas de alta disponibilidad existen los tiempos de fuera de servicio; son mínimos pero existen, van desde 1 minuto o menos hasta 5 o 10 minutos, según sea el caso. En teoría esta es la única diferencia entre ambos, pero en los últimos años, se ha ido acercando la idea de alta disponibilidad a la idea de tolerancia a fallos, debido al abaratamiento de hardware, y de ciertas tecnologías que han ido surgiendo. Estas tecnologías han evolucionado de tal forma, que han logrado que subsistemas donde había que recurrir a la alta disponibilidad ahora, se puede lograr tolerancia a fallos a bajo precio. De todos modos en un sistema (como veremos en la siguiente sección) hay muchos elementos y subsistemas, y algunos subsistemas tolerantes a fallos siguen siendo demasiados caros.

En la mayoría de los análisis, que se hacen de un sistema de servicio, si la aplicación puede estar un mínimo tiempo fuera de servicio, y podemos permitir que el cliente pierda la sesión o la conexión, temporalmente, la alta disponibilidad es una opción muy apropiada. Hay soluciones de alta disponibilidad en las cuales las conexiones se mantienen y las sesiones se recuperan.

### a.2. SPOF (Single Point Of Failure ó punto simple de fallo).

La regla básica para hacer un sistema de alta disponibilidad es la replicación de elementos (Recordando la idea de RAID, Redundant Array of Independent Disks). Con SPOF se quiere hacer referencia a cualquier elemento no replicado y que puede estar sujeto a fallos; afectando con ello al servicio. Se debe evitar el SPOF en cualquier subsistema del sistema, ya que en caso de fallo puede peligrar el servicio de datos por culpa de un subsistema. Por ejemplo, un adaptador de red que da acceso a un servidor a una red es un SPOF, una controladora SCSI también. Si en un entorno de servidores falla uno y no puede ser reemplazado fácilmente por otro, en los tiempos anteriormente mencionados, el servidor se considerará como SPOF.

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

Es conveniente olvidar la redundancia a cierto nivel, pues habrá un nivel en que la alta disponibilidad se hace extremadamente cara, consideremos que sólo tenemos un único edificio para nuestro sistema, perfectamente podemos considerarlo como SPOF (en caso de incendio en el edificio), pero replicar en otro edificio el sistema, se puede salir de presupuesto.

### a.3. Servicio de datos.

El servicio de datos y sus interrupciones es lo que nos ha llevado a adoptar medidas con tolerancia a fallos o alta disponibilidad. En alta disponibilidad, se denomina servicio de datos a un determinado servicio y los recursos necesarios para que el servicio sea ofrecido a un cliente (grupo de recursos). En otros entornos HA, se denominan logical host o software packages a la unión de servicio de datos y grupo de recursos asociados a este.

Estos grupos de recursos han de implementar mecanismos necesarios, para que sean completamente flexibles entre nodos del clúster; es decir puedan ser suplantados o conmutados físicamente entre restantes nodos sin que el

servicio de datos cambie en absoluto. Esta "virtualización" del recurso va a permitir que un nodo pueda suplantar la función de otro dentro del clúster. El servicio de datos sólo será penalizado con el tiempo de conmutación necesario para los grupos de recursos y la puesta en marcha del servicio de datos.

### a.4. Dinámica HA.

Se considera dinámica HA a todas las reconfiguraciones del clúster que garanticen la máxima disponibilidad del servicio de datos. Esta dinámica esta orientada a los nodos integrantes del clúster y la forma en la cual el clúster responde. Se denomina de la siguiente manera:

- **Failover:**

  Es un término genérico que se usa cuando un nodo debe asumir la responsabilidad de otro nodo, importar sus recursos y levantar el servicio de datos. Se ha de entender que una situación de failover es una situación excepcional, para cual la alta disponibilidad ha sido concebida, el fallo de un nodo. Si sólo queda un nodo en el clúster, tras los fallos de los demás, estaremos en un SPOF hasta que el administrador del sistema, verifique y restaure el clúster. También se ha de entender que el servicio de datos sigue levantado, que es el objetivo de la alta disponibilidad.

- **Takeover:**

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

- **Switchover o Giveaway:**

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

- **Splitbrain:**

  Para la gestión de un clúster HA es necesario un mecanismo de comunicación y verificación entre nodos integrantes. Por este mecanismo, cada nodo debe gestionar los recursos que corresponden a cada uno, según el estado del clúster; a su vez cada nodo debe hacer chequeos o latidos (beats) a sus compañeros. Un Splitbrain (división de cerebros) es un caso especial de failover, en el cual falla el mecanismo de comunicación y gestión de un clúster de dos nodos. Es una situación en la cual cada nodo cree que es el único activo, y como no puede saber el estado de su nodo compañero, tomará acciones en consecuencia, forzando un takeover.

**b) Soluciones HA para Linux.**

Con las necesidades anteriormente mencionadas, Linux no podía quedarse atrás. De un tiempo a esta parte han ido surgiendo proyectos y soluciones para Linux, en las cuales algunas destacan su elegancia y sencillez en comparación con sistemas comerciales. En esta sección se va a comentar los mas conocidos y los más utilizados. Se está recibiendo mucha cooperación de distribuciones como VA, SuSE, Red Hat o Conectiva, no debería ni decirse que los esfuerzos de Debian no se quedan atrás.

- ***Heartbeat***

  Es la navaja suiza de la alta disponibilidad para Linux, es una herramienta que permite crear un clúster de alta disponibilidad. De momento el clúster sólo soporta 2 nodos, permite crear grupos de recursos y cambiar estos grupos de recursos fácilmente entre nodos. Carece de herramientas de monitorización del servicio de datos, que se consigue con otras herramientas como mon.

- ***Idirectord y LVS (linux virtual server)***

  LVS permite crear un clúster de balanceo de carga, en el cual hay un nodo que se encarga de gestionar y repartir las conexiones (nodo máster LVS)

entre todos los nodos slave del clúster. El servicio de datos debe residir en todos los nodos slave. LVS puede llegar a soportar sin problemas hasta 200 nodos slave.

ldirectord es un demonio que se ejecuta en el máster LVS, que se encarga de testear el servicio de datos de los nodos slave y eliminarlos e insertarlos en el clúster dinámicamente, si surge algún problema o si se repone el servicio según sea el caso.

- ***Pirahna***

  Es el nombre que Red Hat ha dado a su solución basada en LVS, el añadido es una interfaz para configurarlo.

- ***UltraMonkey***

  Es una solución creada por VA Linux que se basa en LVS y Heartbeat para ofrecer clústers de alta disponibilidad y balanceo de carga. El nodo máster LVS se pone en alta disponibilidad ya que es el único SPOF. Además incorpora una interfaz para configurar el clúster.

- ***Kimberlite***

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

## 2.3 Definición de términos básicos.

i. ***Disponibilidad.*** Cualidad o condición de disponible.

ii. ***Servidor.*** Nodo que, formando parte de una red, provee servicios a otros nodos denominados clientes.

iii. ***Información.*** Conjunto organizado de datos procesados, que constituyen un mensaje que cambia el estado de conocimiento del sujeto o sistema que recibe dicho mensaje.

iv. ***Rendimiento.*** Proporción entre el producto o el resultado obtenido y los medios utilizados.

v. ***Acuerdo a nivel de servicio (SLA).*** Es un contrato escrito entre un proveedor de servicio y su cliente con objeto de fijar el nivel acordado para la calidad de dicho servicio. El ANS es una herramienta que ayuda a ambas partes a llegar a un consenso en términos del nivel de calidad del servicio, en aspectos tales como tiempo de respuesta, disponibilidad horaria, documentación disponible, personal asignado al servicio, etc.

vi. ***Escalabilidad.*** Propiedad deseable de un sistema, una red o un proceso, que indica su habilidad para reaccionar y adaptarse sin perder calidad, o bien manejar el crecimiento continuo de trabajo de manera fluida, o bien para estar preparado para hacerse más grande sin perder calidad en los servicios ofrecidos.

vii. ***Middleware.*** Software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, software, redes, hardware y/o sistemas operativos. Éste simplifica el trabajo de los programadores en la compleja tarea de generar las conexiones que son necesarias en los sistemas distribuidos.

viii. ***Balance de Carga.*** Técnica usada para compartir el trabajo a realizar entre varios procesos, ordenadores, discos u otros recursos. Está íntimamente ligado

a los sistemas de multiprocesamiento, o que hacen uso de más de una unidad de procesamiento para realizar labores útiles.

ix. ***Sistema.*** Objeto complejo cuyos componentes se relacionan con al menos algún otro componente.

x. ***Backus.*** Es una copia de la base de datos en un medio removible.

xi. ***Clúster*** Conjunto de dispositivos (computadoras, equipo de comunicación) que se ven como uno solo, agrupados para dar una mayor disponibilidad y rendimiento a la solución.

xii. ***Downtime.*** Tiempo fuera de servicio planeado o no planeado que pueden experimentar los sistemas al no estar disponible en el caso de fallas en sus componentes.

xiii. ***Failback.*** Proceso de rehabilitar operaciones en el nodo primario cuando ya ha sido reparado.

xiv. ***Failover.*** Proceso por medio del cual un dispositivo electrónico que ha fallado, pasa el control a otro que está en espera de tomarlo.

xv. ***Hardware.*** Toda pieza física y palpable que conforma una computadora.

xvi. ***Host.*** Nombre genérico que se le da a una computadora a la cual se conectan varios usuarios

xvii. ***Mirroring.*** "Espejeado". Copia exacta de un disco a otro.

xviii. ***Nodo.*** Un nodo de un clúster es cualquier sistema que sea miembro de éste.

xix. ***Sistema operativo.*** Conjunto de programas de software que controlan el funcionamiento general del computador y que administran los recursos del mismo.

xx. ***Software.*** Conjunto de programas que pueden ejecutar una computadora.

xxi. ***TCP/IP.*** Protocolo de comunicación utilizado para transmitir datos en una red de computadoras.

xxii. ***Protocolo.*** Conjunto de reglas y métodos por seguir para intercambiar información entre dos computadoras.

xxiii. ***Ruteador.*** Dispositivo electrónico encargado de enviar paquetes de datos basados en las tablas de direcciones físicas de otros dispositivos. Se utilizan para la conexión de diferentes redes de área local.

xxiv. ***Switch.*** Dispositivo electrónico utilizado para conectar dos o más dispositivos en una red de área local. Reconoce la dirección física de los dispositivos, recibe paquetes de datos y selectivamente lo envía por uno de sus puertos.

# CAPÍTULO III. MATERIALES Y MÉTODOS

## 3.1. Desarrollo metodológico.

### 3.1.1. Estado Actual y necesidad de Alta Disponibilidad de la Información.

El caso de estudio se realizó en la empresa MINE SENSE SOLUTIONS que implementará un sistema de Control del Flotas (Despacho de flotas) en el proyecto minero Constancia – Hudbay, para dicha implementación deberá contar con una infraestructura de redes y servidores que garantices un correcto funcionamiento del sistema ya que se trata de un sistema crítico en las operaciones del proyecto.

**a) Características del sistema para el cual se implementara el Clúster de Alta Disponibilidad.**

El Sistema de Control de Flotas tiene con las siguientes características.

- Es una herramienta destinada a mejorar el control y la supervisión de los equipos de campo, utilizando el Sistema de Control de flotas, de equipos de acarreo, carguío y auxiliares, usados frecuentemente en la industria minera para el movimiento de tierra.
- Se puede ejecutar el sistema en ordenadores que usen plataformas Windows, Mac OS o Linux o ser ejecutado en cualquier sistema operativo.
- Su Arquitectura de diseño esta implementada en Cliente-Servidor.
- Utiliza Tecnología WEB, que facilita acceso rápido a los parámetros operativos críticos.
- Desarrollado en el Lenguaje de Programación JAVA.
- Utiliza el gestor de base de datos POSTGRESQL.
- Visualización de la trayectoria de los Equipos de campo.
- Visualización de la velocidad de los equipos de campo.
- Visualización de los estados de actividades, producción y productividad de los equipos (vehículos o maquinaria).

- Trabaja de manera integrada con servidores, Equipos móviles, un Centro de Control, una red de cableado (fibra óptica) e inalámbrica (Wireless).

**b) Necesidad de Alta Disponibilidad en el caso de estudio.**

Las entrevistas realizadas a los desarrolladores, usuarios del sistema, además de expertos en tecnologías mineras dejaron los siguientes escenarios concernientes a la necesidad de la implementación de un clúster de servidores que garantice una alta disponibilidad de la información.

- ✓ La complejidad de las grandes minas de superficie ha sobrepasado por mucho a la capacidad humana para capturar una panorámica exacta del estado operacional en tiempo real por métodos manuales. El simple volumen de datos que emanan de los sistemas de medición y monitoreo de la producción por sí mismo saturaría incluso al modelo más sofisticado de calculadora en segundos.

- ✓ Los sistemas de despacho trabajan con avanzados algoritmos de asignación, gráficas inteligentes y concordancia de software en todo el sistema, mejora la seguridad y brindan una plataforma para futura automatización de equipos.

- ✓ En ninguna parte de las operaciones mineras es más obvio este crecimiento de la dependencia en la recolección, análisis y toma de decisión de datos en un sistema que en las actividades de despacho de flotas, donde los más completos paquetes de programas integrados de gestión de flotas han avanzado bastante más allá de las capacidades básicas de recolección de datos de versiones anteriores y ahora pueden casi ser considerados como sistemas expertos.

- ✓ En una presentación en la 115ª reunión anual de la Northwest Mining Association sostenida a fines de Noviembre del 2009 en Reno, Nevada, EEUU, Desjardins dijo que los sistemas de gestión de flotas no sólo ayuda a los mineros a optimizar la eficiencia en sus activos de carguío y transporte, sino que también se pueden usar como una herramienta para mejorar la seguridad y también como una plataforma para lograr automatización extra en operaciones mineras.

- ✓ Las últimas versiones de los sistemas de gestión de flotas ofrecen algoritmos de asignación altamente sofisticados en combinación con menos características de alta tecnología que realzan el valor del sistema y la utilidad para el cliente; estas incluyen una tendencia hacia el uso de hardware genérico en lugar de diseños propios para redes de comunicaciones, junto con mejores capacidades de almacenamiento de datos integradas para que la información no se pierda si fallan los vínculos de red, y gráficas más inteligentes que hacen un mejor trabajo al evitar que los camiones se "pierdan" en el sistema.

**c) Costo de oportunidad en la utilización de los sistemas de despacho.[24]**

En la industria minera la planificación sustenta la renta posible de capturar a través de la explotación de un recurso natural, según los lineamientos de cada empresa. Las directrices de una operación de extracción, deben ser eficaces y eficientes en la utilización de los recursos con los que se cuenta, de modo de cumplir con los requerimientos de producción y así capturar la renta objetivo. En este esquema, la misión de los sistemas de despacho es servir de herramienta para acercar los procesos de carguío y transporte al nivel óptimo de utilización de recursos, el cual se denomina como la frontera de producción.

En minería a cielo abierto los costos de transporte son cerca de un 50% de los costos de mina, además el proceso de transporte ocupa un 70% del tiempo de un turno, mientras que un 30%, es tiempo no productivo. Lo que habla de un gran potencial a optimizar.

Los beneficios de contar con un sistema de despacho se pueden traducir en beneficios tangibles e intangibles:

- ***Beneficios Tangibles:***

  - Incrementar la productividad de la flota de carguío y transporte dada.
  - Reducir el requerimiento de tamaño de flota asociado a este aumento de productividad.
  - Minimizar el sobre manejo de ciertos equipos.
  - Mezclar distintas restricciones simultáneamente

- Asegurar la velocidad de alimentación a planta.

- ***Beneficios Intangibles:***

  - Pronosticar el rendimiento comparativo entre distintos tipos de equipos de carguío y transporte.
  - Generación de reportes en línea.

**i. Costos de Operación**

Se considerará los insumos básicos de transporte, combustibles y neumáticos por ser estos los insumos de mayor relevancia, pero cabe destacar que al considerar solo estos elementos en el cálculo del costo oportunidad, se subestima su valor, debido a que no se consideran los costos asociados a mantención o insumos menores los cuales se pueden asociar a esta mayor utilización.

***ii.*** **Combustible**

El consumo de combustible por lo general se estima a partir del consumo por hora, pero para este caso se hace necesaria una estimación por distancia por lo que se ha hecho la transformación en base a los datos operativos resultados de una hora operativa estándar. El consumo de un camión CAT 777 es alrededor de 240 litros por hora operativa, considerando que en una hora operativa según los ciclos promedio recorren entre 10 a 16 km, esto sin diferenciar el consumo detenido o en movimiento. Por esto la estimación es compleja de depurar pero se mueve dentro de ciertos rangos.

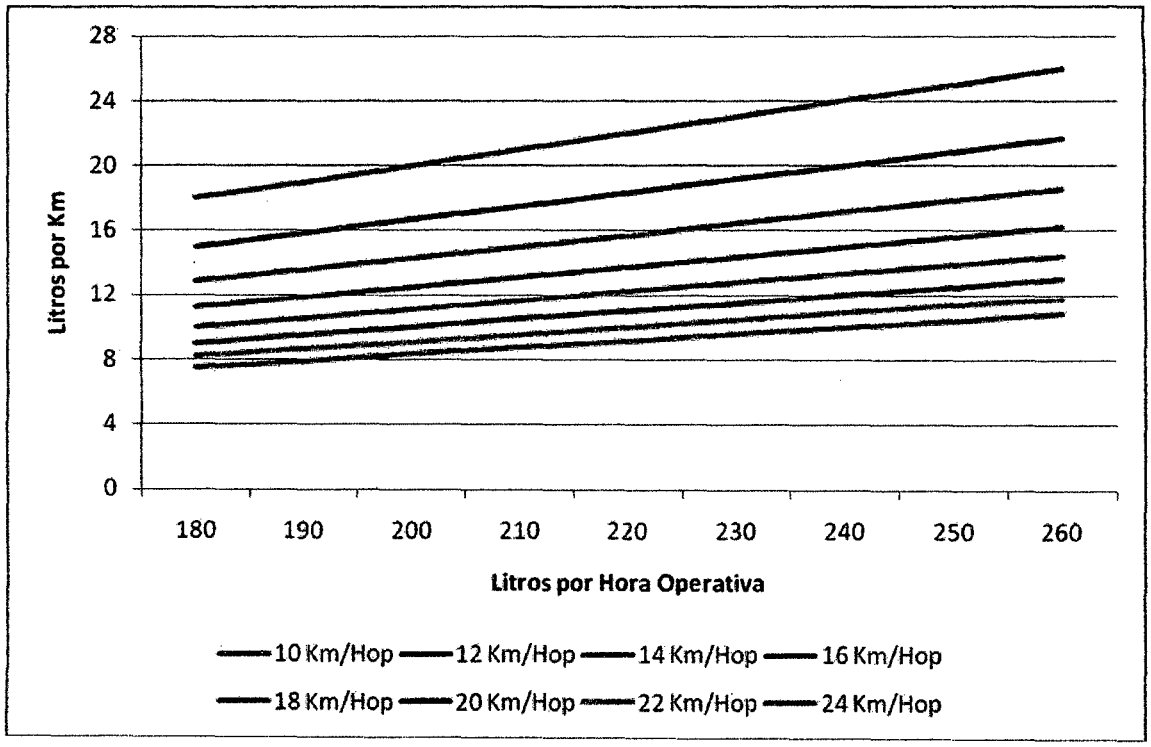



Figura 3. Consumo de Combustible en Operaciones mineras. [24]

### iii. Neumáticos

Para el caso de los neumáticos, el consumo se estima en horas operativas, en el caso estándar cada neumático de CAT 777 (sobre 340ton) se estima una duración de 4200 horas, por lo tanto asumiendo un promedio de 14 Km/Hora, el ciclo de vida en distancia estimado resulta, 4200 Horas Operativas o 175 días Operativos y 58,800 KM.

Los precios de los neumáticos de 3,5 metros (11 pies) de diámetro de los camiones de Caterpillar Inc. que se utilizan para transportar mineral de hierro y carbón rozaron los US$100.000 en el mercado de contado, según Leighton Holdings Ltd., contratista de empresas mineras como BHP Billiton Ltd. y Anglo American Ltd. Los precios llegaron hasta US$150.000 en 2008.[25]

### iv. Herramientas de Toma de Decisión y Diagnóstico

- ***Pantalla de Ruta***

  La pantalla de ruta es la visualización lineal de las rutas entre cada equipo de carguío y sus destinos asociados, según el material y tasas de extracción logradas. Por lo que es muy útil para visualizar el comportamiento de la mina.

- ***Dashboard***

  El Dashboard o tablero, es una herramienta de visualización de indicadores en tiempo real, lo que permite tomar decisiones coherentes con la situación actual de la mina. Con esta herramienta con la cual es posible coordinar la operación de modo de redirigir los indicadores hacia los objetivos del plan.

  El sistema de administración minera también incluye un visualización de los niveles de combustible según lo que puedan indicar los signos vitales del equipo o bien a través de un algoritmo que se ajuste al nivel de consumo de cada equipo y considere la cantidad de litros ingresada al sistema.

- ***Visor de Combustible***

  El sistema de administración minera también incluye un visualización de los niveles de combustible según lo que puedan indicar los signos vitales del equipo o bien a través de un algoritmo que se ajuste al nivel de consumo de cada equipo y considere la cantidad de litros ingresada al sistema.

- ***TKPH***

  La velocidad es uno de los factores que afecta la vida útil de los neumáticos. Para permitir la evaluación del desempeño de los neumáticos en función de temperatura, carga media y velocidad media, fue creado el índice TKPH (Tonelada km por hora). El TKPH es el resultado del producto VM (velocidad media) por CM (carga media), en base diaria, donde el neumático puede operar sin desarrollar temperaturas internas excesivas. El sistema de administración minera (o despacho) posee las herramientas para dar aviso de niveles de riesgo, según los niveles de TKPH establecidos para cada

neumático. Algo que sigue pendiente dentro del mercado de sistemas de despacho, es incluir esta variable dentro de las restricciones de asignación.

**v. Análisis Económico de Resultado.**

Cuando el sistema de optimización de despacho minero propone una configuración, lo hace en base a obtener los tamaños de flota de transporte y las rutas correspondientes, de modo de obtener los mejores resultados en cuando a costo y eficiencia (en la utilización de recursos), además de satisfacer los requerimientos de carguío. Esto significa que no estar en el nivel óptimo implica una diferencia (en costo) que impacta directamente a la renta económica capturada por una planificación, la que sí considera (o debe considerar), niveles efectivos y eficientes de producción.

Esta diferencia significa un costo implícito y asumido por cada turno y que no es considerado en la evaluación anual. Para analizar esta sobre-utilización de recursos se ha recurrido al modelo de costos planteado en el capítulo anterior. Las diferencias en cuanto a la configuración propuesta y la realizada, surgidas de esta metodología son expresadas en distancias de utilización de flota de transporte, lo cual hace posible establecer los costos asociados a recorrer esta diferencia según el modelo de costos definido. La estimación de este costo se realiza en base a utilización de combustible y neumáticos, siendo estos los insumos básicos de la flota de transporte.

**d) Topología y escenario de implementación del Clúster.**

La implementación del Clúster de servidores se realizara en dos zonas geográficas distintas dentro del proyecto minero, las hostiles condiciones climáticas y geográficas que se presentan pueden hacer notar la gran necesidad que se tiene de implementar un Clúster de alta disponibilidad que garantice la continuidad de las operaciones incluso si pierda la totalidad de alguna oficina o del mismo Data Center, la siguiente figura muestra la ubicación geográfica de los 3 servidores que serán parte del clúster.

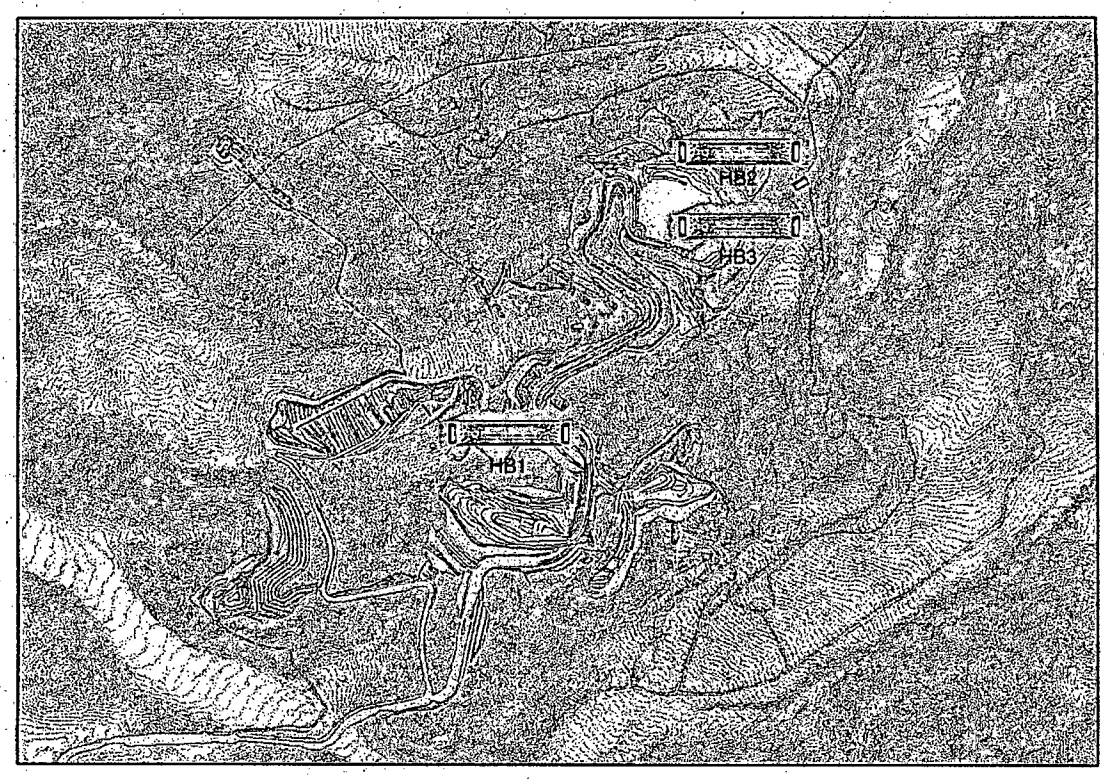



Figura 4. Escenario Geográfico de Implementación del Clúster. [24]

La topología en la que se implementara el Clúster se muestra en la figura 5, muestra cuatro ambientes, Mine Operations, Data Center, Dispach Office y Suport Office, cada uno de estos espacios con un switch que conecta toda la sub red formada en cada ambiente con las demás sub redes de los ambientes, la interconexión entre Mine Operations, Data Center y Dispach Office será en anillo para garantiza que si se pierde algún enlace de fibra, podrá aun tener conexión entre estos ambientes.

Los ambientes de la topología se describen a continuación:

- **MINE OPERATIONS.**

  Mine Operations es el campo de operaciones en donde los equipos móviles (Equipos de carguío, Equipos de acarreo) despliegan sus operaciones, en esta zona se implementara una red del tipo MESH en donde todos los equipos estarán conectados y su vez estos estarán conectados con el servidor que contiene la base de datos de donde obtienen la información que necesitan para actividades y su vez donde almacenan cada una de sus acciones realizadas.

- **DATA CENTER.**

  El Data Center es en donde se encuentran los recursos necesarios para el procesamiento de la información de las operaciones, aquí se encontraran los servidores HB1, HB3 que estarán conectados a un Storage Backup.

- **DISPATCHERS OFFICE.**

  La oficina de despacho o centro de control es en donde se encontrará el servidor HB2 y las PC donde en donde se ejecuta el sistema de control de flotas en donde los Controladores (Personal encargado de la manipulación del sistema) manejan operaciones en campo.

- **SUPPOR OFFICE.**

  La oficina de apoyo es en donde se encontraran los servidores en donde se encontraran los servicios de Share Point y SQL Server (CUBOS OLAP).

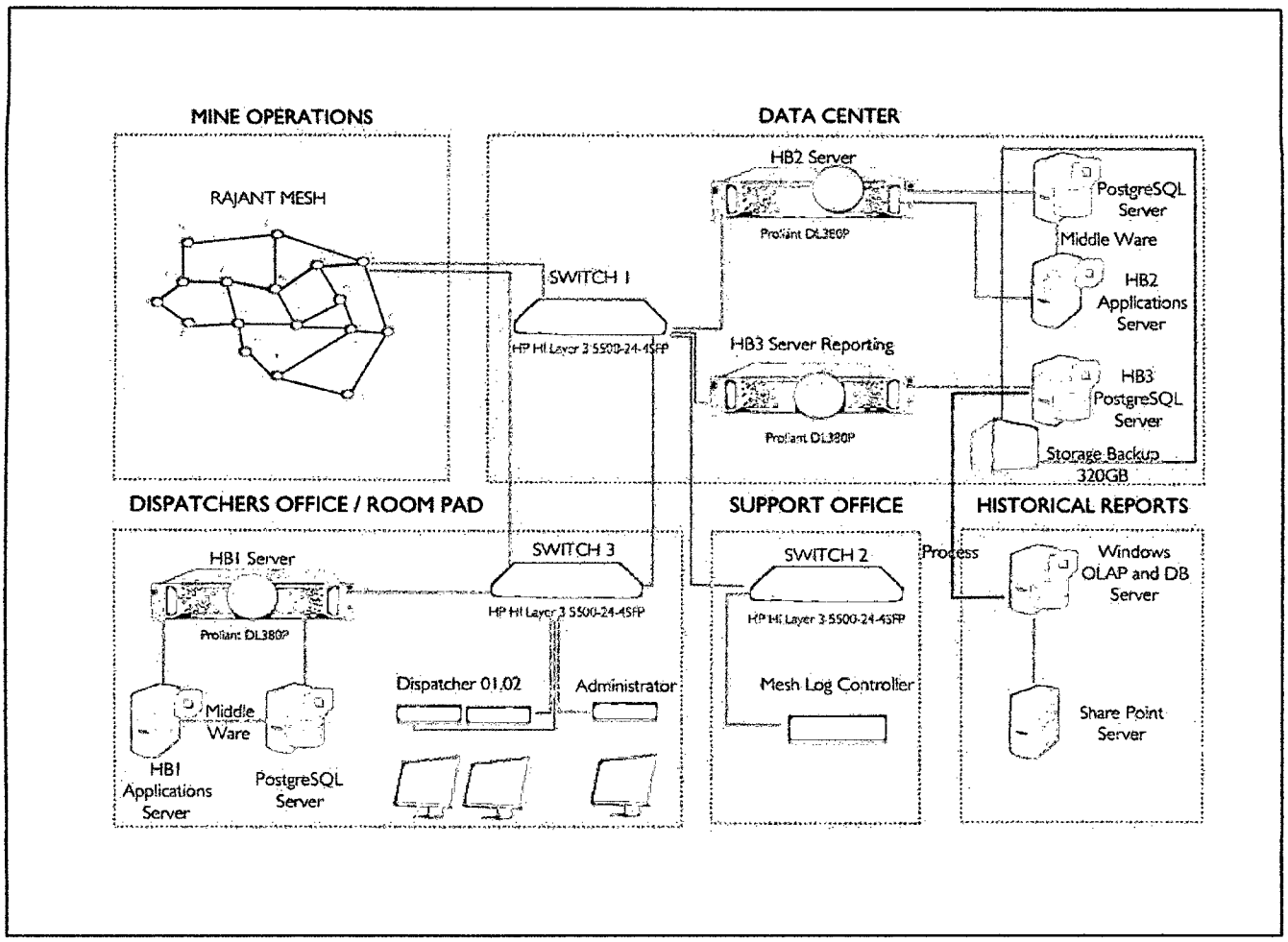



Figura 5. Topología del Clúster. [Fuente propia]

Los servidores que integraran el clúster tendrán instalados los siguientes programas y servicios.

- **Servidor HB1:**

  - ✓ Sistema Operativo Ubuntu Server 12.04 LTS.
  - ✓ Sistema de Gestión de Base de Datos PostgreSQL 9.1
  - ✓ Middleware de Gestión de Base de Datos Pgpool-II.
  - ✓ Demonio de Servicio de Infraestructura de Clúster Heartbeat.
  - ✓ Servidor Web HTTP Apache2.

- **Servidor HB2:**

  - ✓ Sistema Operativo Ubuntu Server 12.04 LTS.
  - ✓ Sistema de Gestión de Base de Datos PostgreSQL 9.1
  - ✓ Middleware de Gestión de Base de Datos Pgpool-II.
  - ✓ Demonio de Servicio de Infraestructura de Clúster Heartbeat.
  - ✓ Servidor Web HTTP Apache2.

- **Servidor HB3:**

  - ✓ Sistema Operativo Ubuntu Server 12.04 LTS.
  - ✓ Sistema de Gestión de Base de Datos PostgreSQL 9.1
  - ✓ Servidor Web HTTP Apache2.

### 3.1.2. Diseño del Clúster.

#### a) Hardware del Clúster.

La descripción de los equipos (Tres Servidores) físicos que conformarán el clúster, un análisis detallado del hardware a utilizar en la implementación del clúster.

El caso de estudio que se ha tomado para la presente tesis tiene disponible proyectado para poner en funcionamiento el clúster equipos con la siguiente tecnología, mostrados en la tabla 2.

| | |
|---|---|
| **MODELO.** | ProLiant DL380p Gen8 2U. |
| **PROCESADOR.** | Intel Xeón Eight Core E5-2650v2, 2.60 GHz |
| **MEM. RAM** | 32 GB DDR3 |
| **DISCO DURO** | HP 600 GB 6G SAS 15k 3.5 in HDD |
| **TARJETA DE RED** | HP Ethernet 1Gb E PCI 331FLR FIO Adptr. |
| **ADICIONALES** | Controlador de Almacenamiento Smart Array P420i/2GB FBWC |
| | Lector Optico HP 12.7mm SATA DVD RW b Kit. |
| | Fuente de Poder Redundante HP 460W CS Platinum |

Tabla 2 .Características del servidor ProLiant DL380p.

Los equipos que conforman el clúster son de gran importancia porque en ellos recaen el funcionamiento adecuado del software que se instale en ellos, si no se cuenta con los equipos adecuados el funcionamiento del clúster no será el deseado, por lo tanto, el hardware es la infraestructura a priori del clúster, por ello, resulta necesario conocer la tecnología con la que se cuenta y aprovechar dichos recursos de la manera más conveniente.

#### b) Software del Clúster.

Ahora se especificará el software a utilizar por los equipos mencionados anteriormente y así como el hardware con el que se dispone es importante, el software a utilizar en dicho hardware es el complemento para sacar el mayor provecho a los recursos con los que se cuenta.

El software a utilizar en la implementación del clúster en su totalidad es de licencia libre, esto quiere decir que está al alcance de cualquier persona que se interese en el enfoque que propone este trabajo de tesis.

| COMPONENTE | VERSIÓN |
|---|---|
| Sistema Operativo | Ubuntu Server 12.04 LTS. |
| Sistema de Gestión de Base de Datos. | PostgreSQL 9.1 |
| Replicación de Base Datos. | Streaming Replication PostgreSQL |
| Demonio de Servicio de Infraestructura de | Heartbeat |
| Protocolo de Sincronización de reloj. | NTP |
| Failover Switch. | Pgpool-II. |
| Load Balancing. | |
| Middleware de Gestión de Base de Datos | |

Tabla 3. Software de los componentes del clúster.

### c) Diseño físico del clúster.

El diseño del clúster cuenta con tres servidores (Servidor Principal, Servidor de Respaldo y Servidor de Reportes), dos de los cuales estarán en clúster y el tercero replicara su información de los dos antes mencionados. En la figura 6 se muestra el arreglo de los componentes del Clúster.

Los Servidores Principal y de Respaldo recibirán las asignaciones del sistema de control y del proceso de optimización alojado en ambos servidores, estas asignaciones serán enviadas a campo donde se encuentran los equipos móviles para realizar sus operaciones de manera correcta, además estos equipos móviles enviarán al servidor todas sus acciones realizadas y la información generada durante toda su labor de manera constante.

Toda la información generada en campo y almacenada en la base de datos de los servidores se verá plasmada a través de gráficos estadísticos a modo de dashboards tanto en los equipos de campo como en el centro de control, esto a través de consultas realizadas constantemente a los servidores que estarán en modo espejo y en balanceo de carga para optimizar el performance y rendimiento óptimo durante los 365 días del año.

El Servidor de Reportes será una copia idéntica de la información contenida en los servidores que conforman el clúster en modalidad de solo lectura, al cual tendrán acceso tanto los reportes históricos (Se implementaran en

servidores complementarios) y los reportes en tiempo real que serán vistos tanto en los centros de operaciones como en cualquier lugar del campo con acceso a red, además tendrán acceso a este servidor el personal encargado de hacer los backups y demás labores administrativas relacionadas con el sistema implementado.

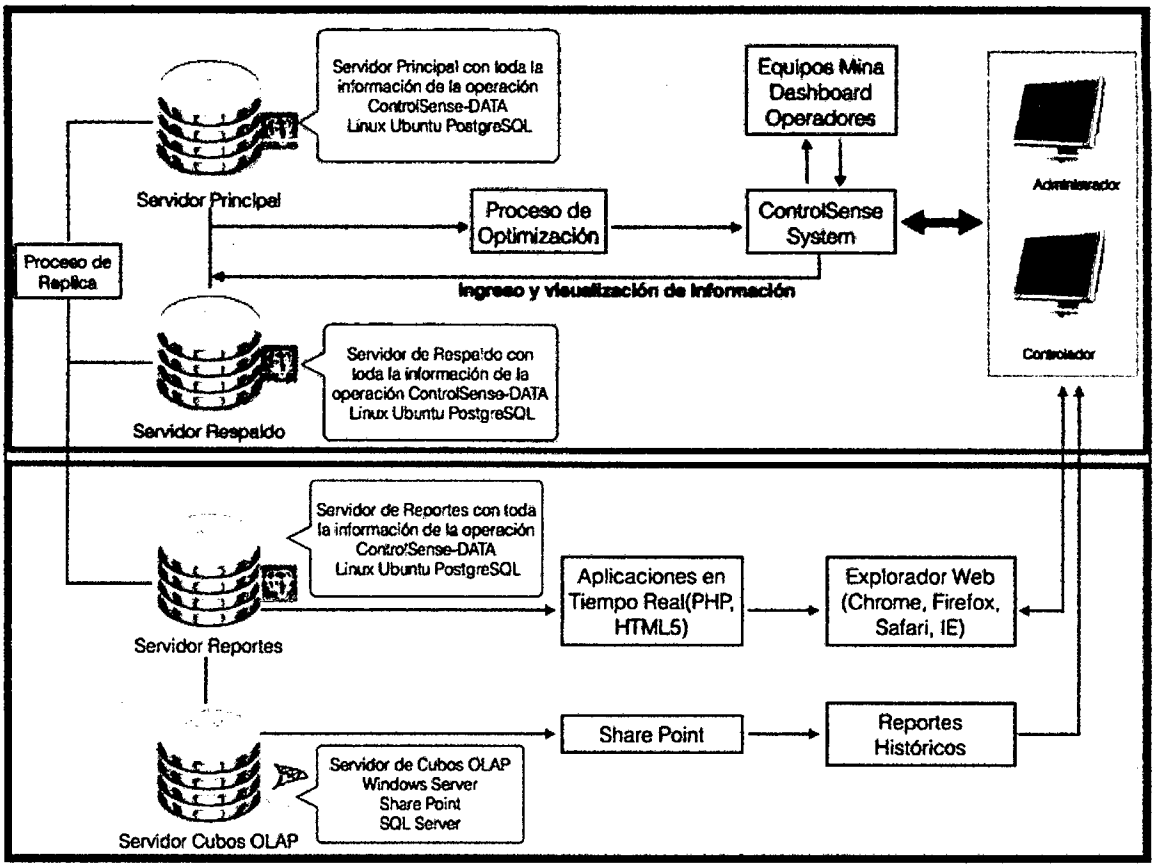



Figura 6. Diseño Físico del Clúster. [Fuente Propia]

### d) Diseño lógico del clúster

El servidor Principal trabaja en Clúster con el Servidor de Respaldo para brindar el soporte de información al sistema, el objetivo del clúster es garantizar alta disponibilidad de la información a través un failover y además distribuir el balance de carga entre los dos servidores que estarán en espejo con una configuración de Streaming Replication. Los Servidores antes mencionados compartirán un IP virtual el cual será al que apuntaran los equipos en campo y el centro de control.

El servidor de Reportes esta fuera del Clúster sin embargo también replica toda la información que le llega al clúster. Este servidor manejara un IP diferente al del Clúster y solo se podrá tener privilegios de lectura mas no de escritura, a este apuntaran únicamente los reportes y determinados usuarios.

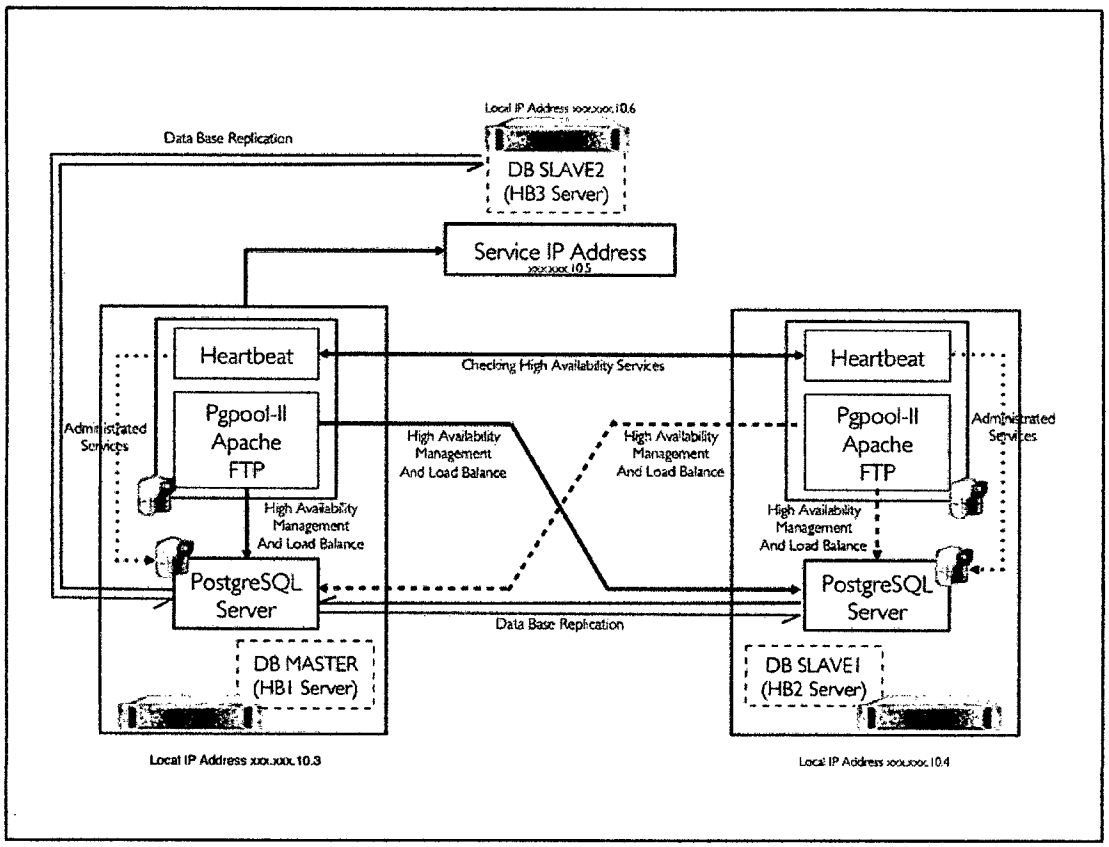



Figura 7. Diseño Lógico del Clúster. [Fuente propia]

## e.1. Replicación.

La replicación es un proceso de intercambio de información a fin de garantizar la coherencia entre los recursos redundantes, como los componentes de software o hardware, para mejorar la confiabilidad, tolerancia a fallos, o la accesibilidad.

La replicación es tema muy importante en cualquier base de datos. En el mundo de la competición de base de datos, PostgreSQL tiene su propia singularidad en RDBMS de código abierto para la alta disponibilidad. PostgreSQL 9.1 tiene soporte incorporado de la replicación sincrónica y asincrónica. La construcción de réplicas asíncronas son Warm Standby, Hot Standby y Streaming Replication y con herramientas de terceros Slony,

londiste, Mammoth etc. La tabla 4 muestra una comparativa entre las herramientas disponibles para el proceso de replicación en PostgreSQL 9.1.

| Herramientas | Replicación Asincrona | | | Replicación Sincrona | | |
|---|---|---|---|---|---|---|
| | Master to Slave | Master to Multiple Slave | Master to Multiple Slave | Master to Slave | Master to Multiple Slave | Master to Multiple Slave |
| Slony-I | SI | SI | *** | *** | *** | *** |
| Bucardo | SI | SI | SI | *** | *** | *** |
| Londiste | SI | SI | *** | *** | *** | *** |
| Mammoth | SI | SI | *** | *** | *** | *** |
| Rubyrep | SI | SI | SI | *** | *** | *** |
| PGCluster | *** | *** | *** | *** | *** | SI |
| pgPool II | *** | *** | *** | *** | SI | SI |
| Streaming Replication | SI | SI | *** | *** | *** | SI |

Tabla 4. Herramientas para el proceso de replicación en PostgreSQL 9.1.

### e.2. Streaming Replication.

Esta característica implementa en el núcleo de PostgreSQL lo necesario para instalar un sistema de replicación asincrónica maestro-esclavo (master-slave) en el que los nodos esclavos se pueden utilizar para realizar consultas de solo lectura. Un sistema de replicación de estas características se podrá usar tanto para añadir redundancia a nuestras bases de datos, como para descargar de trabajo a nuestro servidor principal en lo referente a consultas de solo lectura.

### e.3. Ficheros WAL:

PostgreSQL utiliza los denominados ficheros WAL (Write Ahead Log / REDO) para guardar toda la información sobre las transacciones y cambios realizados en la base de datos. Los ficheros WAL se utilizan para garantizar la integridad de los datos grabados en la base de datos. También se utilizan para reparar automáticamente posibles inconsistencias en la base de datos después de una caída súbita del servidor.

Estos ficheros tienen un nombre único y un tamaño por defecto de 16MB y se generan en el subdirectorio ***pg_xlog*** que se encuentra en el directorio de datos ***($PGDATA)*** usado por PostgreSQL.

El número de ficheros WAL contenidos en ***pg_xlog*** dependerá del valor asignado al parámetro ***checkpoint_segments*** en el fichero de configuración ***postgresql.conf***.

Los ficheros WAL generados en ***pg_xlog*** se reciclan continuamente y en un sistema muy ocupado solo tendremos disponibles en ***pg_xlog*** los últimos cambios ocurridos en la base de datos durante el periodo de tiempo registrado en los ficheros WAL existentes en ***pg_xlog***.

Para activar el archivo automático de ficheros WAL hay que definir los parámetros ***wal_level***, ***archive_mode*** y ***archive_command*** en ***postgresql.conf***. Un fichero WAL se archivará antes que sea reciclado en ***pg_xlog***, pero no antes de que tenga registrado 16MB de información en el mismo.

***Transferencia de registros a nivel de ficheros (file-based log shipping)***: O lo que es lo mismo, transferencia de ficheros WAL completos (16MB de registros) entre servidores de bases de datos.

**Transferencia de registros a nivel de registros (record-based log shipping)**: Transferencia de registros WAL sobre la marcha entre servidores de bases de datos. Esto es lo que hace *Streaming Replication*.

**Replicación/transferencia asincrónica**: Cuando los datos se transfieren de un sistema A a otro B, sin esperar por el *"acuse de recibo"* de B antes de hacer disponibles en A los datos replicados. En un sistema de replicación asincrónico puede existir un cierto retraso ó demora en la disponibilidad de los datos en el sistema esclavo.

**Replicación/transferencia sincrónica**: Cuando los datos se transfieren de un sistema A, a otro B, y A espera el *"acuse de recibo"* de B antes de hacer disponibles en A los datos replicados. En un sistema de replicación sincrónico, todos los datos disponibles en el maestro están disponibles en los esclavos.

La replicación Streaming nos permite transferir asincrónicamente registros WAL sobre la marcha (record-based log shipping) entre un servidor maestro y uno o varios esclavos. Streaming Replication se configura mediante los parámetros **primary_conninfo** en el fichero

`recovery.conf` y `max_wal_senders`, `wal_sender_delay` y `wal_keep_segments` en `postgresql.conf`.

En la práctica un proceso denominado *receptor WAL (WAL receiver)* en el servidor esclavo, se conecta mediante una conexion TCP/IP al servidor maestro. En el servidor maestro existe otro proceso denominado *remitente WAL (WAL sender)* que es el encargado de mandar los registros WAL sobre la marcha al servidor esclavo.

A continuación se tiene un gráfico explicativo de como el Streaming Replication funciona:

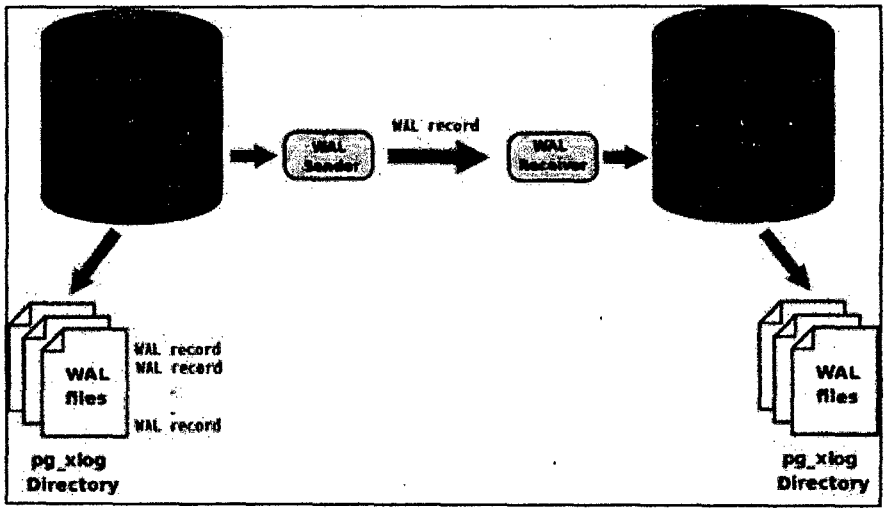



Figura 8. Proceso de Streaming Replication.[26]

### e.4. Sincronización.

La sincronización en tiempo real de los nodos que conforman el *clúster*, es fundamental, porque de ello depende que los datos en los nodos sean los más actuales, por ello es necesario definir la forma en que se establecerá dicha sincronización.

En la sincronización del reloj de cada nodo utilizará el protocolo NTP, la utilización del protocolo asegurará que el reloj de tiempo real sea el mismo en los servidores para evitar generar errores en la sincronización de las réplicas.

En la figura 9 se muestra el arreglo de los nodos del clúster, como servidor NTP primario o *stratum1* se encuentra el nodo maestro que sincroniza su reloj con un servidor público de NTP específico para Sudamerica y en particular el de Perú, entonces con el reloj del nodo maestro sincronizado, este nos servirá para poder sincronizar el reloj de los nodos esclavo 1 y 2 que pertenecerán a un servidor NTP secundario o *stratum 2*, al tener un *stratum* tan corto la sincronización entre los nodos será muy precisa.

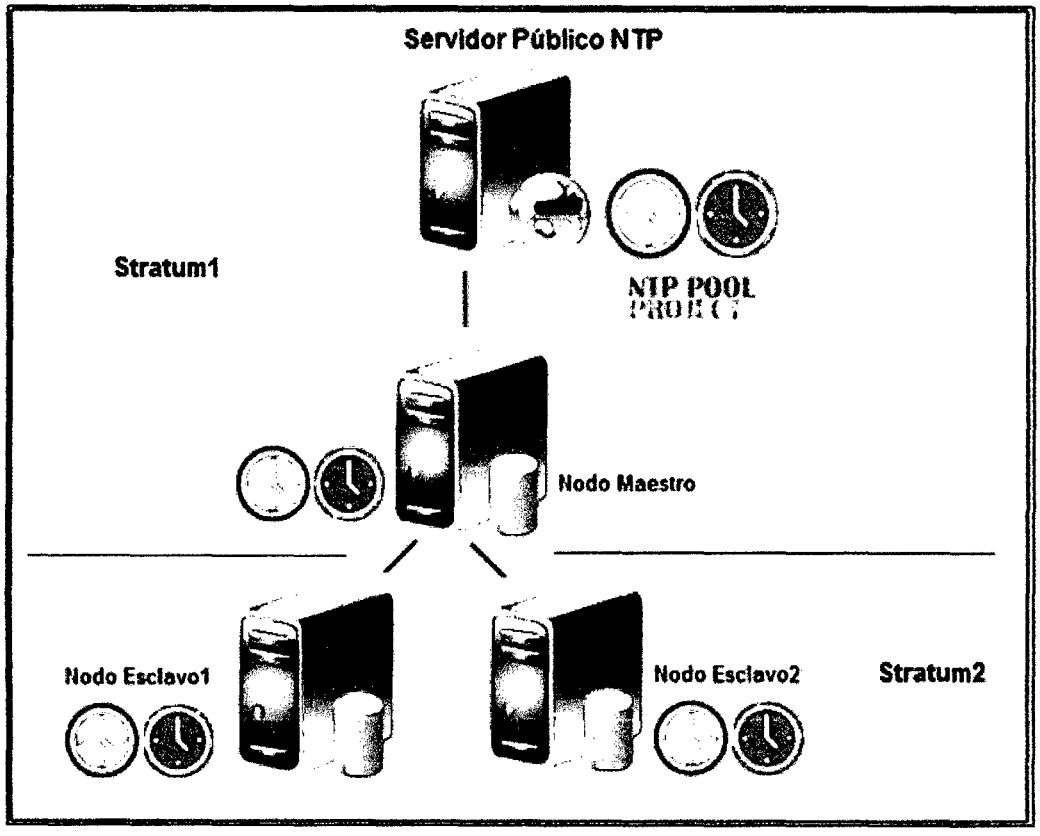



Figura 9. Diseño de sincronización de relojes. [10]

### e.5. Escenario Failover.

En términos generales, el mecanismo de failover permite asegurar la alta disponibilidad de diversos recursos críticos (como el sistema informático del caso de estudio) incluyendo un sistema de respaldo paralelo que se mantiene en ejecución en todo momento, de modo que, en caso de detectarse un fallo en el sistema primario, las tareas a procesar puedan ser automáticamente desviadas hacia el sistema de respaldo, que seguirá dando servicio a los clientes.

Se configurará el clúster en activo-pasivo (primario-secundario), con la salvedad de que se hace uso también del nodo secundario de forma parcial para acelerar las consultas de tipo SELECT (balanceo de carga). El procedimiento de failover en pgpool-II puede suponer tanto un failover hacia el nodo secundario (si ha fallado el primario) como una degeneración del nodo secundario (desactivación del nodo de respaldo).

En la figura 10 se muestra a los nodos que conforman el clúster, el nodo maestro será de lectura y escritura en tanto que los nodos esclavos serán de solo lectura con el objetivo de poder realizar consultas (Balanceo de Carga), la información contenida en el nodo maestro estará en los nodos esclavos al estar configuración de Streaming Replication.

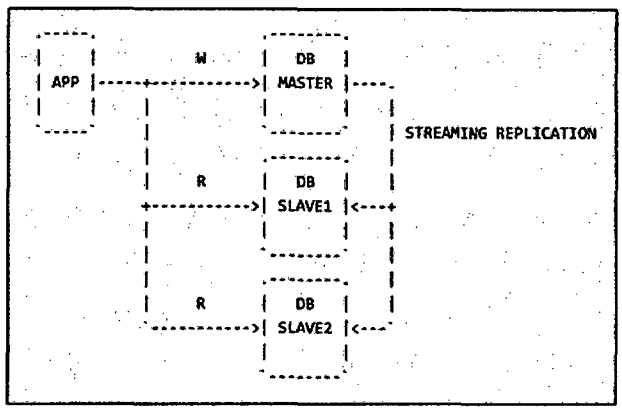



Figura 10. Escenario Failover. [10].

La figura 11 muestra el escenario de caída del nodo maestro y activación del **failover** el cual activara al nodo esclavo 1 como nuevo maestro y pasa a ser nodo de solo Lectura a Lectura/Escritura y al segundo nodo esclavo le cambiara sus parámetros de replicación para que apunte al nuevo nodo maestro.

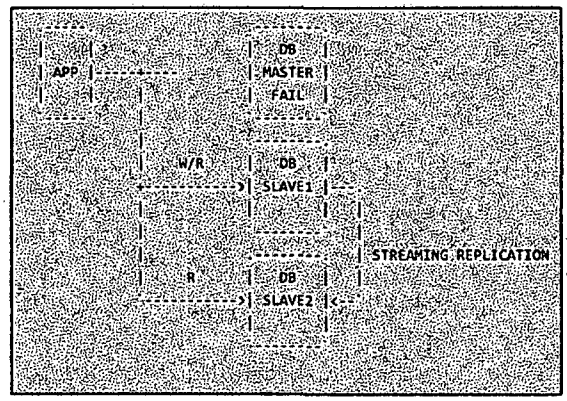



Figura 11. Escenario Failover. [10]

### e.6. Manejo de Clúster y HA (Pgpool - II).

El manejor del CLuster y de la alta disponibilidad que brinda este estará dado por Pgpool-II. Pgpool-II habla backend y frontend protocolo de PostgreSQL, y transmite mensajes entre un backend y un frontend.Por lo tanto, una aplicación de base de datos (frontend) piensa que pgpool-II es el servidor PostgreSQL real, y el servidor (backend) ve pgpool-II como uno de sus clientes. Debido pgpool-II es transparente tanto para el servidor y el cliente, una aplicación de base de datos existente se puede utilizar con pgpool-II casi sin un cambio en el código fuente.

Pgpool-II es un middle ware que se encuentra entre los servidores PostgreSQL y un cliente de base de datos PostgreSQL. Ofrece las siguientes características:

- **Agrupación de conexiones.**

  Pgpool-II mantiene las conexiones establecidas a los servidores PostgreSQL, y los reutiliza cada vez que una nueva conexión con las mismas propiedades (es decir, nombre de usuario, bases de datos, la versión del protocolo) entra en juego. Reduce la conexión de arriba, y mejora el rendimiento general del sistema.

- **Replicación.**

  Pgpool-II puede gestionar múltiples servidores PostgreSQL. Activación de la función de replicación hace que sea posible crear una copia de

seguridad en tiempo real en 2 o más grupos PostgreSQL, de modo que el servicio pueden continuar sin interrupción si falla uno de esos grupos.

- **Equilibrio de carga.**

  Si se replica una base de datos (ya que se ejecuta en cualquiera de los modos de replicación o modo maestro / esclavo), la realización de una consulta SELECT en cualquier servidor devolverá el mismo resultado. pgpool-II se aprovecha de la función de replicación con el fin de reducir la carga en cada servidor PostgreSQL. Lo hace mediante la distribución de consultas SELECT entre los servidores disponibles, mejorando el rendimiento global del sistema. En un escenario ideal, el rendimiento de lectura podría mejorar de forma proporcional al número de servidores PostgreSQL. El equilibrio de carga funciona mejor en un escenario donde hay una gran cantidad de usuarios que ejecutan muchas consultas de sólo lectura al mismo tiempo.

- **Limitar las conexiones exceden.**

  Hay un límite en el número máximo de conexiones simultáneas con PostgreSQL, y las conexiones nuevas se rechaza cuando se alcanza este número. Al aumentar este número máximo de conexiones, sin embargo, aumenta el consumo de recursos y tiene un impacto negativo en el rendimiento general del sistema. pgpool-II también tiene un límite en el número máximo de conexiones, pero las conexiones adicionales se pondrá en cola en lugar de devolver un error de inmediato.

- **Consulta en paralelo.**

  Mediante la función de consulta en paralelo, los datos se pueden dividir entre varios servidores, por lo que una consulta puede ejecutarse en todos los servidores a la vez, reduciendo el tiempo total de ejecución. Consulta en paralelo funciona mejor cuando la búsqueda de datos a gran escala.

Varios modos de funcionamiento están disponibles en pgpool-II. Cada modo tiene asociado características que pueden ser activados o desactivados, y los parámetros de configuración específicos para controlar

sus conductas. El modo de funcionamiento seleccionado para el diseño del clúster a implementar es el Modo Maestro / Esclavo.

| Función / Modo | Modo Raw (* 3) | Modo de replicación | Modo Maestro / Esclavo | Modo de consulta paralelo |
|---|---|---|---|---|
| Connection Pool | X | O | O | O |
| Replicación | X | O | X | (* 1) |
| Equilibrio de carga | X | O | O | (* 1) |
| Failover | O | O | O | X |
| Recuperación en línea | X | O | (* 2) | X |
| Consulta en paralelo | X | X | X | O |
| # Requerida de Servidores | 1 o mayor | 2 o más | 2 o más | 2 o más |
| Requiere Sistema DB? | no | no | no | sí |

- O significa "disponible", X significa 'no disponible'
- (* 1) Modo de consulta paralelo requiere la replicación o el balanceo de carga activados, sin embargo replicación y balanceo de carga no se pueden utilizar para una tabla distribuida en el modo de consulta en paralelo.
- (* 2) la recuperación en línea se puede utilizar con esclavo Maestro + replicación Streaming.
- (* 3) Los clientes sólo tienen que conectar a los servidores PostgreSQL a través pgpool-II. Este modo es útil para simplemente limitando el exceso de conexiones a los servidores, o de activar la conmutación por error con varios servidores.

Tabla 5. Configuración para pgpool-II.

### e.7. HA de Servicio (Heartbeat).

El objetivo fundamental del proyecto Linux-HA es desarrollar una solución de alta disponibilidad (clustering) para Linux que proporcione y promocione la fiabilidad, la disponibilidad y la calidad de servicio o usabilidad (RAS, en sus siglas en inglés) a través del esfuerzo de una comunidad de desarrolladores.

El programa Heartbeat es uno de los componentes principales del proyecto. Fácilmente portable, corre en todos los Linux conocidos, así como en FreeBSD y Solaris. Heartbeat es una de las implementaciones principales del estándar Open Clúster Framework (OCF).

Heartbeat fue la primera pieza de software que se escribió para el proyecto Linux-HA. Puede llevar a cabo la detección de la caída de nodos, las comunicaciones y la gestión del clúster en un solo proceso. Actualmente soporta un modelo de dependencias muy sofisticado para clústeres de N nodos, y es muy útil y estable. La unidad de gestión de Heartbeat es el

clúster incluso si no hay servicios ejecutándose. Para la mayoría de sistemas, una dirección de este tipo es obligatoria.

Asimismo, se recomienda que se reserve una de estas direcciones para cada interfaz, de modo que se puedan testear las interfaces incluso cuando no estén activas.

### 3.1.3. Implementación del Clúster.

En este capítulo se describen detalladamente las etapas de implementación del *Clúster*, con las configuraciones necesarias para lograr su correcto funcionamiento, esto implica desde la instalación del sistema operativo hasta la implementación de plan de contingencia ante la caída de algunos de los nodos que conforman el *Clúster*, con la finalidad de dejar claro al lector los pasos y pruebas que se realizaron en el mismo.

El siguiente subcapítulo nos guiará a través de los pasos necesarios para implementar cada uno de los componentes del clúster.

#### a. Instalación y configuración del sistema operativo.

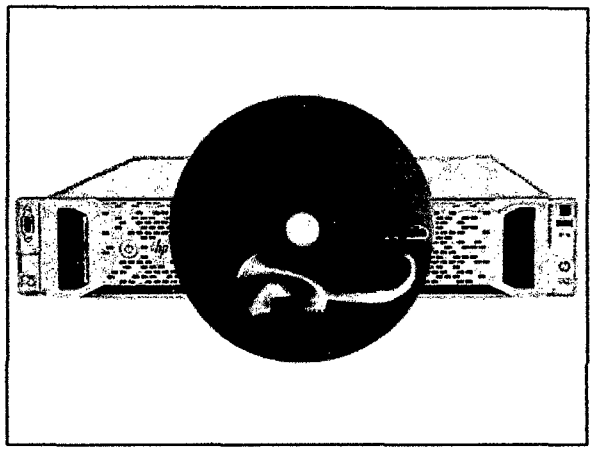


Figura 12. Disco de instalación de SO Ubuntu Server 12.04.

La primera etapa consiste en la instalación del sistema operativo en los 3 servidores. A continuación se explica el proceso detalladamente.

Para instalar Ubuntu server necesitamos seguir los siguientes pasos:

recurso. Los recursos pueden ser, por ejemplo, direcciones IP o servicios (aplicaciones). Los siguientes tipos de aplicaciones son típicos ejemplos:

- Servidores de bases de datos.
- Servidores web.
- Aplicaciones ERP.
- Servidores de correo electrónico.
- Cortafuegos.
- Servidores de ficheros.
- Servidores de DNS.
- Servidores de DHCP.
- Servidores de proxy-caché.

Un recurso es la unidad básica de la alta disponibilidad. Un recurso es un servicio o facility el cuál pasa a tener alta disponibilidad mediante el gestor de recursos del clúster de alta disponibilidad.

Un recurso es una abstracción que puede ser de diferentes tipos. Puede ser algo muy concreto, como un volumen de disco o un lector de tarjetas, o puede ser más abstracto, como una dirección IP, un conjunto de reglas de firewall o un servicio de software (como un servidor web o un servidor de base de datos).

- **start**: iniciar o adquirir el control del recurso.
- **stop**: finalizar o ceder el control del recurso.
- **status**: consultar si el recurso está iniciado o parado.
- **monitor**: consultar de manera más detallada si el recurso está operando correctamente.

Nótese que los recursos del tipo R1 (versión 1 de Linux-HA) deben soportar status, mientras que los del tipo R2 (versión 2) deben soportar la operación monitor.

El gestor de recursos del clúster de alta disponibilidad intenta conseguir que todos los recursos estén disponibles para los usuarios asegurándose de que estén ejecutándose en alguno de los nodos del clúster.

El gestor de recursos de Heartbeat R1 (y muchos otros gestores de recursos en clúster) aúna diversos recursos en grupos, llamados

ResourceGroups. En ese caso, cada grupo es iniciado, detenido o movido en su conjunto por el gestor de recursos del clúster.

Los ficheros de configuración de Heartbeat están en /etc/ha.d. Necesitamos crear tres ficheros:

- **ha.cf**: fichero de configuración principal.
- **haresources**: fichero de configuración de recursos.
- **authkeys**: información de autenticación.

Llegados a este punto, es preciso introducir dos nuevos conceptos de uso frecuente con Heartbeat, dirección IP de servicio y dirección IP de administración.

Una dirección de servicio es una dirección que es gestionada por el sistema de HA, y que es movida por el clúster allí donde los servicios correspondientes se estén ejecutando. Estas direcciones de servicio son direcciones a través de las cuales los clientes y usuarios de los servicios en HA acceden a dichos servicios. Típicamente se almacenan en DNS con nombres conocidos.

Es importante que la dirección de servicio no sea gestionada por el sistema operativo, sino que sea el software de HA el único que la maneje. Si se le da una dirección administrativa al sistema de HA para que la gestione, ésto causará problemas pues se confundirá al sistema de HA y el sistema operativo y el sistema de HA se pelearán por el control de esta dirección.

El agente de recursos IPaddr2 es capaz de levantar una interfaz desde cero, incluso si no se ha establecido ninguna dirección base (la versión anterior necesitaba de una dirección base en cualquier interfaz, pues tan sólo era capaz de añadir o eliminar direcciones a interfaces ya levantados). Además, no existe límite en el número de direcciones IP por interfaz que IPaddr2 puede gestionar. En cambio, una dirección administrativa es una dirección que está permanentemente asociada a un nodo específico del clúster.

Tales direcciones son muy útiles, y se recomienda encarecidamente que una dirección de este tipo sea reservada para cada nodo del clúster, de manera que el administrador de sistemas pueda acceder al nodo del

- ✓ Descargar Ubuntu Server 12.04 LTS y quemar el *software* en un cd o USB booteable, disponible en: **http://www.ubuntu.com/**

- ✓ Colocar el cd/USB de instalación y ***bootear*** desde él, ejecutarlo y responder a las siguientes preguntas dentro el proceso de instalación de Ubuntu server:

  ***Lenguaje***: English.

  ***Ubicación***: Perú.

  ***Hostname***: hb1

  ***Nombre del dominio***: mss1.hbm.hb1

- ✓ El instalador pregunta sobre la partición del disco, se elige el método guiado donde se crean las particiones automáticas, la de sistema (/) y la de ***swap***.

- ✓ Se configura usuarios y contraseñas, entonces se le proporciona contraseña al administrador y al usuario común.

- ✓ La instalación prosigue y finalmente el sistema ha sido instalado.

- ✓ Finalmente después de reiniciar el equipo podremos entrar al sistema (figuras 13 y 14):

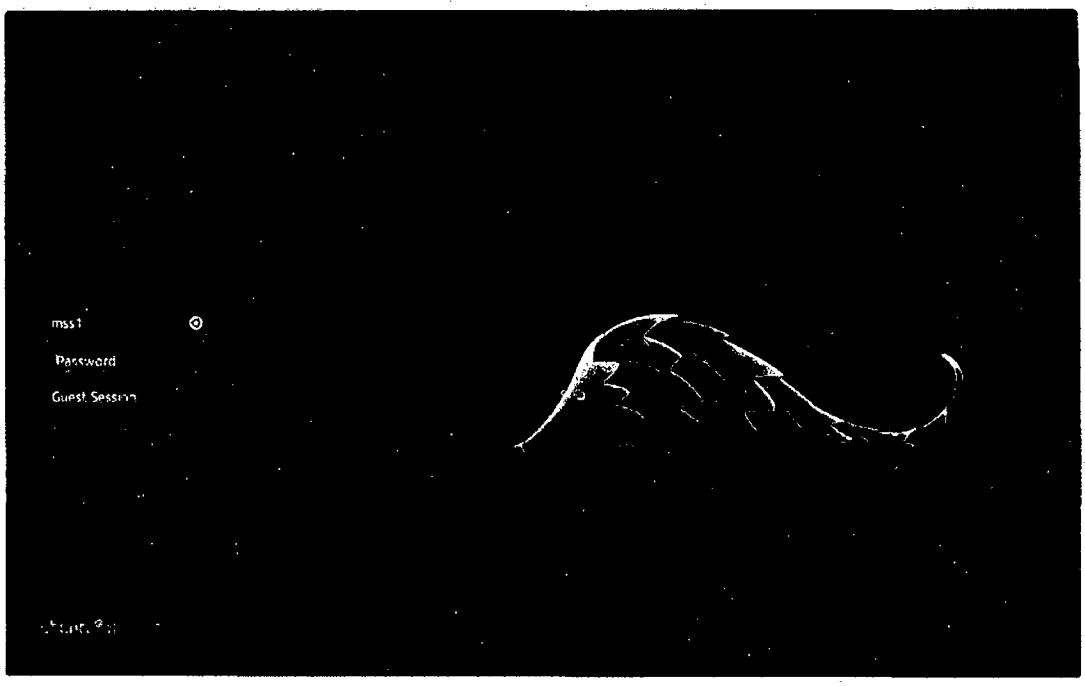



Figura 13. Pantalla de logueo de Ubuntu Server 12.04 LTS. [Fuente propia]

Figura 14. Entorno Gráfico de Ubuntu Server 12.04 LTS. [Fuente propia]

Una vez instalado el OS se necesita configurar la dirección IP estática. Para cambiar la configuración de red es necesario modificar dos archivos con los siguientes comandos:

```
sudo gedit /etc/resolv.conf
```

En el archivo `resolv.conf` agregamos las siguientes líneas:

```
nameserver xxx.xxx.xx.xxx (Ejemplo 192.167.70.141)
```

Guardamos y luego ejecutamos en siguiente comando en la terminal:

```
sudo gedit /etc/network/interfaces
```

En el archivo interfaces agregaremos: address, netmask, network, broadcast, gateway, dns-nameservers, quedando de la siguiente manera:

```
# This file describes the network interfaces available on
your system
# and how to activate them. For more information, see
interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#  auto eth0
#  iface eth0 inet dhcp
    auto eth0
    iface eth0 inet static
      address 192.167.70.xxx
      netmask 255.255.255.0
      network 192.167.70.0
      broadcast 192.167.70.255
      gateway 192.167.70.1
      dns-nameservers 192.167.70.xxx
```

Luego removemos el DHCP-Client con el siguiente comando:

```
sudo apt-get remove dhcp-client
```

Finalmente restamos los networking components:

```
sudo   /etc/init.d/networking
```

La configuración del servidor ya está completa ahora solo hay que reiniciar el SO.

**b. Configuración PostgreSQL.**

✓ **Configuración SSH.**

SSH (Secure SHell, en español: intérprete de órdenes segura) es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquinas remotas a través de una red. Permite manejar por completo la computadora mediante un intérprete de comandos, y también puede redirigir el tráfico de X para poder ejecutar programas gráficos si tenemos ejecutando un Servidor X (en sistemas Unix y Windows).

Además de la conexión a otros dispositivos, SSH nos permite copiar datos de forma segura (tanto archivos sueltos como simular sesiones FTP cifradas), gestionar claves RSA para no escribir claves al conectar a los dispositivos y pasar los datos de cualquier otra aplicación por un canal seguro tunelizado mediante SSH.

Todos los nodos del Clúster deberán estar habilitados para el acceso remoto de los demás, la configuración SSH se dará en la instancia de postgresql.

Acceso a la instancia postgresql:

```
sudo su - postgres
```

Generar SSH key:

```
ssh-keygen -t rsa -P ""
```

Enviar SSH key a nodos:

```
scp ~/.ssh/id_rsa.pub 192.167.70.xx:~/.ssh/id_rsamss1.pub
```

Autorizar el servidor SSH para utilizar las claves públicas.

```
cat id_rsamss1.pub >> authorized_keys
```

**c. Configuración de cada nodo para Streaming Replication.**

✓ **Configuración del nodo maestro/esclavo.**

La configuración se realizara en dos de los archivos de configuración por defecto del postgres ubicadas en la ruta `/etc/postgresql/9.1/main`.

```
Sudo gedit /etc/postgresql/9.1/main/postgresql.conf
```

```
listen_addresses = '*'
wal_level = hot_standby
max_wal_senders = 2
wal_keep_segments = 50
hot_standby = on
archive_mode = on

archive_command = 'test ! -f
/var/lib/postgresql/9.1/archiving_active || cp -i %p
/var/lib/postgresql/9.1/archive/%f
```

Se deberán tener dos archivos adicionales que servirán de respaldo para el archivo de configuración `postgresql.conf`:

(`postgresql.conf.master`, `postgresql.conf.slave`)

```
# -----------------------------
# PostgreSQL configuration file
# -----------------------------
#
# This file consists of lines of the form:
#
#   name = value
#
# (The "=" is optional.)  Whitespace may be used.  Comments are introduced with
# "#" anywhere on a line.  The complete list of parameter names and allowed
# values can be found in the PostgreSQL documentation.
#
# The commented-out settings shown in this file represent the default values.
# Re-commenting a setting is NOT sufficient to revert it to the default value;
# you need to reload the server.
#
# This file is read on server startup and when the server receives a SIGHUP
# signal.  If you edit the file on a running system, you have to SIGHUP the
# server for the changes to take effect, or use "pg_ctl reload".  Some
# parameters, which are marked below, require a server shutdown and restart to
# take effect.
#
# Any parameter can also be given as a command-line option to the server, e.g.,
# "postgres -c log_connections=on".  Some parameters can be changed at run time
# with the "SET" SQL command.
#
# Memory units:  kB = kilobytes        Time units:  ms  = milliseconds
#                MB = megabytes                 s   = seconds
#                GB = gigabytes                 min = minutes
#                                               h   = hours
#                                               d   = days
#------------------------------------------------------------------------------
# FILE LOCATIONS
#------------------------------------------------------------------------------

# The default values of these variables are driven from the -D command-line
# option or PGDATA environment variable, represented here as ConfigDir.

data_directory = '/var/lib/postgresql/9.1/main'# use data in another directory
# (change requires restart)
hba_file = '/etc/postgresql/9.1/main/pg_hba.conf'# host-based authentication file
# (change requires restart)
ident_file = '/etc/postgresql/9.1/main/pg_ident.conf'# ident configuration file
# (change requires restart)

# If external_pid_file is not explicitly set, no extra PID file is written.
external_pid_file = '/var/run/postgresql/9.1-main.pid'# write an extra PID file
# (change requires restart)
```

```
#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------
# - Connection Settings -

listen_addresses = '*'          # what IP address(es) to listen on;
# comma-separated list of addresses;
# defaults to 'localhost', '*' = all
# (change requires restart)
port = 5432# (change requires restart)
max_connections = 100# (change requires restart)
# Note:  Increasing max_connections costs ~400 bytes of shared memory per
# connection slot, plus lock space (see max_locks_per_transaction).
#superuser_reserved_connections = 3# (change requires restart)
unix_socket_directory = '/var/run/postgresql'# (change requires restart)
#unix_socket_group = ''# (change requires restart)
#unix_socket_permissions = 0777# begin with 0 to use octal notation
# (change requires restart)
#bonjour = off# advertise server via Bonjour
# (change requires restart)
#bonjour_name = ''# defaults to the computer name
# (change requires restart)
# - Security and Authentication -
#authentication_timeout = 1min# 1s-600s
ssl = true# (change requires restart)
#ssl_ciphers = 'ALL:!ADH:!LOW:!EXP:!MD5:@STRENGTH'# allowed SSL ciphers
# (change requires restart)
#ssl_renegotiation_limit = 512MB# amount of data between renegotiations
password_encryption = on
#db_user_namespace = off
# Kerberos and GSSAPI
#krb_server_keyfile = ''
#krb_srvname = 'postgres'# (Kerberos only)
#krb_caseins_users = off
# - TCP Keepalives -
# see "man 7 tcp" for details
#tcp_keepalives_idle = 0# TCP_KEEPIDLE, in seconds;
# 0 selects the system default
#tcp_keepalives_interval = 0# TCP_KEEPINTVL, in seconds;
# 0 selects the system default
#tcp_keepalives_count = 0# TCP_KEEPCNT;
# 0 selects the system default
#------------------------------------------------------------------------------
# RESOURCE USAGE (except WAL)
#------------------------------------------------------------------------------
# - Memory -

shared_buffers = 24MB# min 128kB
# (change requires restart)
#temp_buffers = 8MB# min 800kB
#max_prepared_transactions = 0# zero disables the feature
# (change requires restart)
# Note:  Increasing max_prepared_transactions costs ~600 bytes of shared memory
# per transaction slot, plus lock space (see max_locks_per_transaction).
# It is not advisable to set max_prepared_transactions nonzero unless you
# actively intend to use prepared transactions.
#work_mem = 1MB# min 64kB
#maintenance_work_mem = 16MB# min 1MB
#max_stack_depth = 2MB# min 100kB
# - Kernel Resource Usage -
#max_files_per_process = 1000# min 25
# (change requires restart)
#shared_preload_libraries = ''# (change requires restart)
# - Cost-Based Vacuum Delay -

#vacuum_cost_delay = 0ms# 0-100 milliseconds
#vacuum_cost_page_hit = 1# 0-10000 credits
#vacuum_cost_page_miss = 10# 0-10000 credits
#vacuum_cost_page_dirty = 20# 0-10000 credits
#vacuum_cost_limit = 200# 1-10000 credits
# - Background Writer -

#bgwriter_delay = 200ms# 10-10000ms between rounds
#bgwriter_lru_maxpages = 100# 0-1000 max buffers written/round
#bgwriter_lru_multiplier = 2.0# 0-10.0 multipler on buffers scanned/round
```

```
#------------------------------------------------------------------------------
# WRITE AHEAD LOG
#------------------------------------------------------------------------------
# - Settings -

wal_level = hot_standby# minimal, archive, or hot_standby
# (change requires restart)
#fsync = on# turns forced synchronization on or off
#synchronous_commit = on# synchronization level; on, off, or local
#wal_sync_method = fsync# the default is the first option
# supported by the operating system:
#   open_datasync
#   fdatasync (default on Linux)
#   fsync
#   fsync_writethrough
#   open_sync
#full_page_writes = on# recover from partial page writes
#wal_buffers = -1# min 32kB, -1 sets based on shared_buffers
# (change requires restart)
#wal_writer_delay = 200ms# 1-10000 milliseconds

#commit_delay = 0# range 0-100000, in microseconds
#commit_siblings = 5# range 1-1000
# - Checkpoints -
#checkpoint_segments = 3# in logfile segments, min 1, 16MB each
#checkpoint_timeout = 5min# range 30s-1h
#checkpoint_completion_target = 0.5# checkpoint target duration, 0.0 - 1.0
#checkpoint_warning = 30s# 0 disables

# - Archiving -

archive_mode = on# allows archiving to be done
# (change requires restart)

archive_command = 'test ! -f /var/lib/postgresql/9.1/archiving_active || cp -i %p
/var/lib/postgresql/9.1/archive/%f'

# command to use to archive a logfile segment
#archive_timeout = 0# force a logfile segment switch after this
# number of seconds; 0 disables


#------------------------------------------------------------------------------
# REPLICATION
#------------------------------------------------------------------------------

# - Master Server -

# These settings are ignored on a standby server

max_wal_senders = 2# max number of walsender processes
# (change requires restart)
#wal_sender_delay = 1s# walsender cycle time, 1-10000 milliseconds
wal_keep_segments = 50# in logfile segments, 16MB each; 0 disables
#vacuum_defer_cleanup_age = 0# number of xacts by which cleanup is delayed
#replication_timeout = 60s# in milliseconds; 0 disables
#synchronous_standby_names = ''# standby servers that provide sync rep
# comma-separated list of application_name
# from standby(s); '*' = all

# - Standby Servers -

# These settings are ignored on a master server

hot_standby = on# "on" allows queries during recovery
# (change requires restart)
#max_standby_archive_delay = 30s# max delay before canceling queries
# when reading WAL from archive;
# -1 allows indefinite delay
#max_standby_streaming_delay = 30s# max delay before canceling queries
# when reading streaming WAL;
# -1 allows indefinite delay
#wal_receiver_status_interval = 10s# send replies at least this often
# 0 disables
#hot_standby_feedback = off# send info from standby to prevent
# query conflicts
```

```
#------------------------------------------------------------------------------
# QUERY TUNING
#------------------------------------------------------------------------------

# - Planner Method Configuration -

#enable_bitmapscan = on
#enable_hashagg = on
#enable_hashjoin = on
#enable_indexscan = on
#enable_material = on
#enable_mergejoin = on
#enable_nestloop = on
#enable_seqscan = on
#enable_sort = on
#enable_tidscan = on
# - Planner Cost Constants -
#seq_page_cost = 1.0# measured on an arbitrary scale
#random_page_cost = 4.0# same scale as above
#cpu_tuple_cost = 0.01# same scale as above
#cpu_index_tuple_cost = 0.005# same scale as above
#cpu_operator_cost = 0.0025# same scale as above
#effective_cache_size = 128MB

# - Genetic Query Optimizer -
#geqo = on
#geqo_threshold = 12
#geqo_effort = 5# range 1-10
#geqo_pool_size = 0# selects default based on effort
#geqo_generations = 0# selects default based on effort
#geqo_selection_bias = 2.0# range 1.5-2.0
#geqo_seed = 0.0# range 0.0-1.0
# - Other Planner Options -

#default_statistics_target = 100# range 1-10000
#constraint_exclusion = partition# on, off, or partition
#cursor_tuple_fraction = 0.1# range 0.0-1.0
#from_collapse_limit = 8
#join_collapse_limit = 8# 1 disables collapsing of explicit
# JOIN clauses
#------------------------------------------------------------------------------
# RUNTIME STATISTICS
#------------------------------------------------------------------------------

# - Query/Index Statistics Collector -

#track_activities = on
#track_counts = on
#track_functions = none# none, pl, all
#track_activity_query_size = 1024 # (change requires restart)
#update_process_title = on
#stats_temp_directory = 'pg_stat_tmp'

# - Statistics Monitoring -

#log_parser_stats = off
#log_planner_stats = off
#log_executor_stats = off
#log_statement_stats = off

#------------------------------------------------------------------------------
# AUTOVACUUM PARAMETERS
#------------------------------------------------------------------------------

#autovacuum = on# Enable autovacuum subprocess?  'on'
# requires track_counts to also be on.
#log_autovacuum_min_duration = -1# -1 disables, 0 logs all actions and
# their durations, > 0 logs only
# actions running at least this number
# of milliseconds.
#autovacuum_max_workers = 3# max number of autovacuum subprocesses
# (change requires restart)
#autovacuum_naptime = 1min# time between autovacuum runs
#autovacuum_vacuum_threshold = 50# min number of row updates before
# vacuum
```

```
#autovacuum_analyze_threshold = 50# min number of row updates before
# analyze
#autovacuum_vacuum_scale_factor = 0.2# fraction of table size before vacuum
#autovacuum_analyze_scale_factor = 0.1# fraction of table size before analyze
#autovacuum_freeze_max_age = 200000000# maximum XID age before forced vacuum
# (change requires restart)
#autovacuum_vacuum_cost_delay = 20ms# default vacuum cost delay for
# autovacuum, in milliseconds;
# -1 means use vacuum_cost_delay
#autovacuum_vacuum_cost_limit = -1# default vacuum cost limit for
# autovacuum, -1 means use
# vacuum_cost_limit

#------------------------------------------------------------------------------
# CLIENT CONNECTION DEFAULTS
#------------------------------------------------------------------------------
# - Statement Behavior -
#search_path = '"$user",public'# schema names
#default_tablespace = ''# a tablespace name, '' uses the default
#temp_tablespaces = ''# a list of tablespace names, '' uses
# only default tablespace
#check_function_bodies = on
#default_transaction_isolation = 'read committed'
#default_transaction_read_only = off
#default_transaction_deferrable = off
#session_replication_role = 'origin'
#statement_timeout = 0# in milliseconds, 0 is disabled
#vacuum_freeze_min_age = 50000000
#vacuum_freeze_table_age = 150000000
#bytea_output = 'hex'# hex, escape
#xmlbinary = 'base64'
#xmloption = 'content'
# - Locale and Formatting -
datestyle = 'iso, mdy'
#intervalstyle = 'postgres'
#timezone = '(defaults to server environment setting)'
#timezone_abbreviations = 'Default'     # Select the set of available time zone
# abbreviations.  Currently, there are
#   Default
#   Australia
#   India
# You can create your own file in
# share/timezonesets/.
#extra_float_digits = 0# min -15, max 3
#client_encoding = sql_ascii# actually, defaults to database
# encoding

# These settings are initialized by initdb, but they can be changed.
lc_messages = 'en_US.UTF-8'# locale for system error message
# strings
lc_monetary = 'en_US.UTF-8'# locale for monetary formatting
lc_numeric = 'en_US.UTF-8'# locale for number formatting
lc_time = 'en_US.UTF-8'# locale for time formatting

# default configuration for text search
default_text_search_config = 'pg_catalog.english'

# - Other Defaults -

#dynamic_library_path = '$libdir'
#local_preload_libraries = ''

#------------------------------------------------------------------------------
# LOCK MANAGEMENT
#------------------------------------------------------------------------------

#deadlock_timeout = 1s
#max_locks_per_transaction = 64# min 10
# (change requires restart)
# Note:  Each lock table slot uses ~270 bytes of shared memory, and there are
# max_locks_per_transaction * (max_connections + max_prepared_transactions)
# lock table slots.
#max_pred_locks_per_transaction = 64# min 10
# (change requires restart)
```

```
#------------------------------------------------------------------------------
# VERSION/PLATFORM COMPATIBILITY
#------------------------------------------------------------------------------

# - Previous PostgreSQL Versions -

#array_nulls = on
#backslash_quote = safe_encoding# on, off, or safe_encoding
#default_with_oids = off
#escape_string_warning = on
#lo_compat_privileges = off
#quote_all_identifiers = off
#sql_inheritance = on
#standard_conforming_strings = on
#synchronize_seqscans = on

# - Other Platforms and Clients -

#transform_null_equals = off


#------------------------------------------------------------------------------
# ERROR HANDLING
#------------------------------------------------------------------------------

#exit_on_error = off# terminate session on any error?
#restart_after_crash = on# reinitialize after backend crash?


#------------------------------------------------------------------------------
# CUSTOMIZED OPTIONS
#------------------------------------------------------------------------------

#custom_variable_classes = ''# list of custom variable class names
```

En el mismo fichero en el cual se encuentra el archivo de configuración `postgresql.conf` se encuentra el archivo de configuración `pg_hba.conf` en el cual configuraremos los accesos al servicio de base de datos postgres.

```
host  Replication   repl  192.167.70.X1/32   md5
host  replication   repl  192.167.70.X2/32   md5

host  all     pgpool  192.167.70.X4/32   md5
host  all     postgres  192.167.70.X4/32   trust

host  all     pgpool  192.167.70.X5/32   md5
host  all     postgres  192.167.70.X5/32   trust

host  all     prominesys  192.167.70.X4/32   trust
```

Ahora se crearan los usuarios para la replicación y acceso del middleware.

```
master# /etc/init.d/postgresql start
master# su - postgres
postgres@master$ psql
```

```
postgres=# CREATE USER repl REPLICATION ENCRYPTED PASSWORD 'repl';
postgres=# CREATE USER pgpool LOGIN ENCRYPTED PASSWORD 'pgpool';
```

Creamos el base backup:

```
master# su - postgres

postgres@master$ cd /var/lib/postgresql/9.1/
postgres@master$ mkdir archive
postgres@master$ touch archiving_active
postgres@master$ psql -c "select
pg_start_backup('base_backup');"
postgres@master$ tar -cvf base_backup.tar --exclude=pg_xlog -
-exclude=postmaster.pid main/
postgres@master$ psql -c "select pg_stop_backup();"
postgres@master$ tar -rf base_backup.tar archive
postgres@master$ rm archiving_active
postgres@master$ scp base_backup.tar slave1:~
postgres@master$ scp base_backup.tar slave2:~
```

Incluir el archivo de recuperación en el fichero `main`:

```
postgres@slave1$ vim main/recovery.done
standby_mode = 'on'
primary_conninfo = 'host=192.167.70.11 port=5432 user=repl
password=repl'
restore_command = 'cp /var/lib/postgresql/9.1/archive/%f %p'
recovery_target_timeline='latest'
trigger_file = '/tmp/pgsql.trigger'
```

- ✓ **Streaming-Replication Script:**

Este Script se ejecutara conjuntamente con el script de recuperación online-recovery.

```
#!/bin/bash
##This is meanst to be run on the slave, with the masters ip as the passed
variable. ($1)
sourcehost="$1"
datadir=/var/lib/postgresql/9.1/main
archivedir=/var/lib/postgresql/9.1/archive
archivedirdest=/var/lib/postgresql/9.1/archive
#Usage
if [ "$1" = "" ] || [ "$1" = "-h" ] || [ "$1" = "-help" ] || [ "$1" = "--
help" ];
then
echo "Usage: $0 masters ip address"
exit 0
fi
#This script must be run as postgres user
Whoami () {
    if [[ $(whoami) != "postgres" ]]
    then
        echo "[INFO] This script must be run as postgres user !"
exit 1
    fi
}
#Check if postgres server is running on remote host
CheckIfPostgresIsRunningOnRemoteHost () {
isrunning="$(ssh  postgres@"$1"  'if  killall  -0  postgres;  then  echo
"postgres_running"; else echo "postgress_not_running"; fi;')"

if [[ "$isrunning" = "postgress_not_running" ]]
then
echo "[ERROR] Postgres not running on the master. Exiting..";
       exit 1

elif [[ "$isrunning" = "postgres_running" ]]
then
echo "[OK] Postgres master running on remote host";

elif echo "[ERROR] Unexpected response. Exiting.."
then
exit 1
fi
}
#Check if the supposed master is actually a master
CheckIfMasterIsActuallyAMaster () {
    ismaster="$(ssh          postgres@"$1"          'if          [          -f
/var/lib/postgresql/9.1/main/recovery.done         ];         then         echo
"postgres_is_a_master_instance"; else echo "postgres_is_not_master"; fi;')"
    if [[ "$ismaster" = "postgres_is_not_master" ]]
    then
        echo "[ERROR] Postgres is already running as a slave. Exiting..";
        exit 1
    elif [[ "$ismaster" = "postgres_is_a_master_instance" ]]
    then
        echo "[INFO] Postgres is running as master (probably)";
    elif echo "[ERROR] Unexpected response. Exiting.."
    then
        exit 1
    fi
}
```

```
#prepare local server to become the new slave server.
PrepareLocalServer () {

    if [ -f '/tmp/pgsql.trigger' ]
    then
            rm /tmp/pgsql.trigger
    fi
    echo "[INFO] Stopping slave node.."
    bash /etc/init.d/postgresql stop
    if [[ -f "$datadir/recovery.done" ]];
    then
            mv "$datadir"/recovery.done "$datadir"/recovery.conf
    fi
    #Remove old WAL logs
    rm /var/lib/postgresql/9.1/archive/*
}
CheckForRecoveryConfig () {
    if [[ -f "$datadir/recovery.conf" ]];
    then
        echo "[OK] Slave config file found, Continuing.."
    else
        echo "[ERROR] recovery.conf not found. Postgres is not a slave.
Exiting.."
        exit 1
    fi
}
#Put master into backup mode
#Before doing PutMasterIntoBackupMode clean up archive logs (IE rm or mv
/var/lib/postgresql/9.1/archive/*). They are not needed since we are
effectivly createing a new base backup and then synching it.
PutMasterIntoBackupMode () {
    echo "[INFO] Putting postgres master '$1' in backup mode."
    ssh postgres@"$1" "rm /var/lib/postgresql/9.1/archive/*"
    ssh postgres@"$1" "psql -c \"SELECT pg_start_backup('Streaming
Replication', true)\" postgres"
}

#rsync master's data to local postgres dir
RsyncWhileLive () {
    echo "[INFO] Transfering data from master '$1' ..."
    rsync -C -av --delete --progress -e ssh --exclude server.key --exclude
server.crt --exclude recovery.conf --exclude recovery.done --exclude
postmaster.pid --exclude pg_xlog/ "$1":"$datadir"/ "$datadir"/ > /dev/null
    if [ $? == 0 ]
    then
        echo "[OK] Transfert completed.";
    else
echo "[ERROR] Error during transfer !";
exit 0;
    fi
}
#This archives the WAL log (ends writing to it and moves it to the $archive
dir
StopBackupModeAndArchiveIntoWallLog () {
    echo "[INFO] Disable backup mode from master '$1'."
    ssh postgres@"$1" "psql -c \"SELECT pg_stop_backup()\" postgres"
    echo "[INFO] Synchronising master/slave archive directory..."
    rsync -C -a --progress -e ssh "$1":"$archivedir"/ "$archivedirdest"/ >
/dev/null
    if [ $? == 0 ]
    then
        echo "[OK] Sync achieved.";
    else
        echo "[ERROR] Error during sync !";
        exit 0;
    fi
}
```

```
#stop postgres and copy transactions made during the last two rsync's
StopPostgreSqlAndFinishRsync () {
    echo "[INFO] Stopping master node.."
    ssh postgres@"$1" "/etc/init.d/postgresql stop"
    echo "[INFO] Transfering xlog files from master... "
    rsync -av --delete --progress -e ssh "$sourcehost":"$datadir"/pg_xlog/
"$datadir"/pg_xlog/ > /dev/null
    if [ $? == 0 ]
    then
        echo "[OK] Transfert completed.";
    else
        echo "[ERROR] Error during transfer !";
        exit 0;
    fi
}

#Start both Master and Slave
StartLocalAndThenRemotePostGreSql () {
    echo "[INFO] Starting slave node.."
    /etc/init.d/postgresql start
    if ! killall -0 postgres; then echo '[ERROR] Slave not running !'; else
echo "[OK] Slave started."; fi;


    echo "[INFO] Starting master node.."
    ssh postgres@"$1" "/etc/init.d/postgresql start"

    status=$(ssh  postgres@$1 "if ! killall -0 postgres; then echo 'error';
else echo 'running'; fi;")
    if [ $status == "error" ]
    then
        echo "[ERROR] Master not running !";
        exit 0;
    else
        echo "[OK] Master started.";
    fi
}

#Execute above operations
Whoami
CheckIfPostgresIsRunningOnRemoteHost "$1"
CheckIfMasterIsActuallyAMaster "$1"
PrepareLocalServer "$datadir"
CheckForRecoveryConfig "$datadir"
PutMasterIntoBackupMode "$1"
RsyncWhileLive "$1"
StopBackupModeAndArchiveIntoWallLog "$1" "$archivedir" "$archivedirdest"
StopPostgreSqlAndFinishRsync "$1"
StartLocalAndThenRemotePostGreSql "$1"
```

- ✓ **Configuración del nodo esclavo / maestro.**

La configuración se realizara en dos de los archivos de configuración por defecto del postgres ubicadas en la ruta `/etc/postgresql/9.1/main`.

```
Sudo gedit /etc/postgresql/9.1/main/postgresql.conf
```

```
listen_addresses = '*'
wal_level = hot_standby
max_wal_senders = 2
wal_keep_segments = 50
hot_standby = on
```

Se deberán tener dos archivos adicionales que servirán de respaldo para el archivo de configuración postgresql.conf (postgresql.conf.master, postgresql.conf.slave).

En el mismo fichero en el cual se encuentra el archivo de configuración `postgresql.conf` se encuentra el archivo de configuración `pg_hba.conf` en el cual configuraremos los accesos al servicio de base de datos postgres.

```
host  replication   repl          192.167.70.10/32   md5
host  replication   repl          192.167.70.12/32   md5

host  all           pgpool        192.167.70.14/32   md5
host  all           postgres      192.167.70.14/32   trust

host  all           pgpool        192.167.70.15/32   md5
host  all           postgres      192.167.70.15/32   trust

host  all           prominesys    192.167.70.14/32   trust
```

Replicación:

```
slave1# su - postgres
  postgres@slave1$ cd /var/lib/postgresql/9.1
  postgres@slave1$ mv ~/base_backup.tar .
  postgres@slave1$ rm -rf main/
  postgres@slave1$ tar -xvf base_backup.tar
  postgres@slave1$ mkdir main/pg_xlog
  postgres@slave1$ vim main/recovery.conf
```

Archivo de recuperación.

```
  standby_mode = 'on'
  primary_conninfo = 'host=192.167.70.10 port=5432 user=repl
password=repl'
  restore_command = 'cp /var/lib/postgresql/9.1/archive/%f %p'
  recovery_target_timeline='latest'
  trigger_file = '/tmp/pgsql.trigger'
```

- ✓ **Streaming-Replication Script:**

Este Script se ejecutara conjuntamente con el script de recuperación *online-recovery*.

**Configuración del nodo esclavo.**

La configuración se realizara en dos de los archivos de configuración por defecto del postgres ubicadas en la ruta /etc/postgresql/9.1/main.

```
Sudo gedit /etc/postgresql/9.1/main/postgresql.conf
```

```
listen_addresses = '*'
wal_level = hot_standby
max_wal_senders = 2
wal_keep_segments = 50
hot_standby = on
```

En el mismo fichero en el cual se encuentra el archivo de configuración `postgresql.conf` se encuentra el archivo de configuración `pg_hba.conf` en el cual configuraremos los accesos al servicio de base de datos postgres.

```
host     all  pgpool     192.167.70.1x/32  md5
host     all  postgres   192.167.70.1x/32  trust

host     all  pgpool     192.167.70.1x/32  md5
host     all  postgres   192.167.70.1x/32  trust

host     all  prominesys 192.167.70.1x/32  trust
```

Replicación:

```
slave2# su - postgres

  postgres@slave2$ cd /var/lib/postgresql/9.1

  postgres@slave2$ mv ~/base_backup.tar .
```

Archivo de recuperación:

```
standby_mode = 'on'
primary_conninfo = 'host=192.167.70.10 port=5432 user=repl
password=repl'
restore_command = 'cp /var/lib/postgresql/9.1/archive/%f %p'
recovery_target_timeline='latest'
```

**d. Configuración de Pgpool-II.**

El middleware pgpool2 se deberá instalar con el siguiente comando de terminal.

```
apt-get -y install pgpool2 postgresql-9.1-pgpool2
```

La configuración de pgpool2 se realizara en consola con los siguientes comandos.

```
app# cd /etc/pgpool2/
app# cp pgpool.conf{,.back}
app# cp pcp.conf{,.back}
app# echo pgpool:`pg_md5 pgpool` >> pcp.conf
app# vim pgpool.conf
```

El archivo de configuración principal de pgpool2 presenta los siguientes parámetros.

```
#-------------#
# CONNECTIONS #
#-------------#

listen_addresses = '*'
port = 5432
socket_dir = '/var/run/postgresql'
pcp_port = 9898
pcp_socket_dir = '/var/run/postgresql'
#---------#
# BACKEND #
#---------#
backend_hostname0='192.167.70.x'
backend_port0 = 5432
backend_weight0 = 0
backend_data_directory0                                    =
'/var/lib/postgresql/9.1/main'
backend_flag0 = 'ALLOW_TO_FAILOVER'

backend_hostname1='192.167.70.x'
backend_port1 = 5432
backend_weight1 = 1
backend_data_directory1                                    =
'/var/lib/postgresql/9.1/main'
backend_flag1 = 'ALLOW_TO_FAILOVER'
```

```
#------#
# AUTH #
#------#
enable_pool_hba = on
pool_passwd = 'pool_passwd'
authentication_timeout = 60
#-----#
# SSL #
#-----#
ssl = off
#-------#
# POOLS #
#-------#
num_init_children = 100
max_pool = 4
child_life_time = 1
child_max_connections = 2
connection_life_time = 1
client_idle_limit = 300

#------#
# LOGS #
#------#
log_destination = 'stderr'
print_timestamp = on
log_connections = on
log_hostname = off
log_statement = on
log_per_node_statement = on
log_standby_delay = 'none'
syslog_facility = 'LOCAL0'
syslog_ident = 'pgpool'
debug_level = 0

#----------------#
# FILE LOCATIONS #
#----------------#

pid_file_name = '/var/run/postgresql/pgpool.pid'
logdir = '/var/log/postgresql'

#--------------------#
# CONNECTION POOLING #
#--------------------#

connection_cache = on
reset_query_list = 'ABORT; DISCARD ALL'
```

```
#------------------#
# REPLICATION MODE #
#------------------#
replication_mode = off
replicate_select = off
insert_lock = on
lobj_lock_table = ''
replication_stop_on_mismatch = off
failover_if_affected_tuples_mismatch = off


#---------------------#
# LOAD BALANCING MODE #
#---------------------#
load_balance_mode = on
ignore_leading_white_space = on
white_function_list = ''
black_function_list                                              =
'nextval,setval,a.*,b.*,c.*,d.*,e.*,f.*,g.*,h.*,i.*,j.*,
k.*,l.*,m.*,n.*,o.*,p.*,q.*,r.*,s.*,t.*,u.*,v.*,w.*,x.*,
y.*,z.*'

#-------------------#
# MASTER/SLAVE MODE #
#-------------------#
master_slave_mode = on
master_slave_sub_mode = 'stream'

sr_check_period = 0
sr_check_user = 'pgpool'
sr_check_password = 'pgpool'
delay_threshold = 0
follow_master_command = ''

#------------------------------#
# PARALLEL MODE AND QUERY CACHE #
#------------------------------#
parallel_mode = off
enable_query_cache = off
pgpool2_hostname = ''
system_db_hostname  = 'localhost'
system_db_port = 5432
system_db_dbname = 'pgpool'
system_db_schema = 'pgpool_catalog'
system_db_user = 'pgpool'
system_db_password = ''
```

```
system_db_password = ''

#--------------#
# HEALTH CHECK #
#--------------#

health_check_period = 0
health_check_timeout = 20
health_check_user = 'pgpool'
health_check_password = 'pgpool'

#-----------------------#
# FAILOVER AND FAILBACK #
#-----------------------#

failover_command  =  '/var/lib/postgresql/bin/failover.sh
%d %M %m'
failback_command = ''
fail_over_on_backend_error = on

#-----------------#
# ONLINE RECOVERY #
#-----------------#

recovery_user = 'nobody'
recovery_password = ''
recovery_1st_stage_command = ''
recovery_2nd_stage_command = ''
recovery_timeout = 90
client_idle_limit_in_recovery = 0


#--------#
# OTHERS #
#--------#
relcache_expire = 0
```

✓ **Failover Script:**

```
app# su - postgres
  postgres@app$ mkdir bin
  postgres@app$ vim bin/failover.sh
```

Se contara con dos scripts de failover uno para el caso en el que el servidor maestro sea el que se configura de tal manera desde el inicio, y el otro para cuando el primer esclavo ascienda como maestro.

Se tendrá que dar permisos `chmod` para la ejecución de dichos scripts que deberán ser creados con el comando VIM.

```
postgres@app$ chmod u+x bin/failover.sh
```

Failover Script:

```
#!/bin/sh -
  FALLING_NODE=$1
  OLD_MASTER=$2
  NEW_MASTER=$3
  SLAVE1='192.167.70.248'
  SLAVE2='192.167.70.252'
  if test $FALLING_NODE -eq 0
  then

    ssh -T $SLAVE1 touch /tmp/pgsql.trigger
    echo "SLAVE1 R/W TRANSACTION"

    ssh -T $SLAVE1 "while test ! -f
/var/lib/postgresql/9.1/main/recovery.done; do sleep 1;
done; scp /var/lib/postgresql/9.1/main/pg_xlog/*history*
$SLAVE2:/var/lib/postgresql/9.1/main/pg_xlog/"

    echo "SLAVE1 IS NEW MASTER"

    ssh -T $SLAVE2 "sed -i
's/192.167.70.249/192.167.70.248/'
/var/lib/postgresql/9.1/main/recovery.conf"
    ssh -T $SLAVE2 /etc/init.d/postgresql restart

    echo "SLAVE2 --> NEW MASTER"

    /usr/sbin/pcp_attach_node 10 localhost 9898 pgpool
pgpool 2

    echo "FAIL OVER END"
  fi
```

✓ **Online-recovery Script:**

Este Script se ejecutara en el caso de que el nodo maestro caiga y se tenga que incluirlo nuevamente en el Clúster pero como esclavo y el esclavo que se encuentra en el papel de maestro, lo tomara también en la configuración del pgpool2 para mejorar el rendimiento del Clúster en el balance carga.

Online-recovery Script:

```
#!/bin/bash
#Postgres data directory
postgres_datadir='/var/lib/postgresql/9.1/main'
#Postgres configuration directory
postgres_configdir='/etc/postgresql/9.1/main'
#Postgres user ssh key
postgres_user_key='/var/lib/postgresql/.ssh/id_rsa'
#Pgpool configuration directory
pgpool_configdir='/etc/pgpool2'

if [ -f '/tmp/postgres_master' ]
then
    #Get current postgres master id
    current_master_id=$(cat /tmp/postgres_master);
else
    echo "[ERROR] /tmp/postgres_master not found !";
    exit 0;
fi

#Get postgres master name
current_master_name=$(pcp_node_info 10 localhost 9898 pgpool
pgpool $current_master_id | cut -d' ' -f1)
#Get postgres slave id
[  $current_master_id  ==  0  ]  &&  current_slave_id=1  ||
current_slave_id=0
#Get postgres slave name
current_slave_name=$(pcp_node_info 10 localhost 9898 pgpool
pgpool $current_slave_id | cut -d' ' -f1)

#Test if pgpool is running
CheckIfPgpoolIsRunning () {
    #Send signal 0 to pgpool to check if it's running
    if ! killall -0 pgpool; then echo "[ERROR] Pgpool is not
running !"; exit 1; fi;
}
```

```
AttachNodeToPgpool () {
   #pcp_attach_node is a command that permit to attach a
specific postgres server (identified by 6th parameter) to
pgpool.
   #pcp_attach_node dont return a good error code when it
fails so here if I catch "BackendError" message in stderr I
presume
   #that attachment failed.
   #TODO:find a condition to break the folowing loop if
attachment fails.
   while [ "`pcp_attach_node 10 localhost 9898 pgpool pgpool
$1`" == "BackendError" ]
    do
        pcp_attach_node 10 localhost 9898 pgpool pgpool  $1;
        #This sleep is recommanded to avoid stressing pgpool
in this infinite loop.
        sleep 5;
    done
}

#Whether the slave node is down, start it and attach it to
pgpool's backend pool.
ReattachDegeneratedSlave () {
    #Reboot slave node
    echo "[INFO] Slave node '$current_slave_name' is down.
Performing postgres server reboot..."
    #Remote postgres reboot via ssh
    ssh  -i  $postgres_user_key  postgres@$current_slave_name
"/etc/init.d/postgresql restart"
    #Test if postgres is running
    status=$(ssh             -i             $postgres_user_key
postgres@$current_slave_name "if ! killall -0 postgres; then
echo 'error'; else echo 'running'; fi;")

if [ $status == "error" ]
    then
        echo "[ERROR] Postgres slave still down !";
        exit 0;
    else
        echo "[OK] Slave node successfully started.";
    fi

    #Do 'slave online recovery' to force slave sync if it
has incoherent data relatevely to master.
```

```
#echo "[INFO] Starting online recovery for slave
'$current_slave_name' ..."
    #ssh -i /var/lib/postgresql/.ssh/id_rsa
postgres@$current_slave_name "bash
/var/lib/postgresql/streaming-replication.sh
$current_master_name"
    #Atttach slave (even master) to pgpool's backends pool
    #Reattach the master node if you have performed an
online recovery for slave node and not juste a simple
reboot.
    #Attempting to reatach master to pgpool's backend pool
    #echo "[INFO] Attaching master node
'$current_master_name' ..."
    #AttachNodeToPgpool "$current_master_id"
    #echo "[OK] Master node '$current_master_name' has been
successfully reattached to pgpool."
    #Attempting to reattach slave to pgpool's backend pool
    echo "[INFO] Attaching slave node
'$current_slave_name'..."
    AttachNodeToPgpool "$current_slave_id"
    echo "[OK] Slave node '$current_slave_name' has been
successfully reattached to pgpool."
}

#Whether the master is down do the folowing operations :
SwitchOldMasterToSlave () {

    new_master_name=$current_slave_name
    new_master_id=$current_slave_id
    new_slave_name=$current_master_name
    new_slave_id=$current_master_id
    #Setup old master config to slave mode
    echo "[INFO] Setting up configuration for the new slave
node '$new_slave_name'..."
    ssh -i $postgres_user_key postgres@$new_slave_name
"/etc/init.d/postgresql stop"
    ssh -i $postgres_user_key postgres@$new_slave_name "cp -
p $postgres_configdir/postgresql.conf.slave
$postgres_configdir/postgresql.conf"
    ssh -i $postgres_user_key postgres@$new_slave_name "[ -
f $postgres_datadir/recovery.done ] && mv
$postgres_datadir/recovery.done
$postgres_datadir/recovery.conf"
# Switch slave to new master
    echo "[INFO] Setting up configuration for the new master
'$new_master_name'..."
```

```
ssh -i $postgres_user_key postgres@$new_master_name "[ -f
/tmp/pgsql.trigger ] && rm /tmp/pgsql.trigger"
    ssh -i $postgres_user_key postgres@$new_master_name "[ -
f      $postgres_datadir/recovery.conf      ]      &&      mv
$postgres_datadir/recovery.conf
$postgres_datadir/recovery.done"
    ssh -i $postgres_user_key postgres@$new_master_name "cp
-p                $postgres_configdir/postgresql.conf.master
$postgres_configdir/postgresql.conf"
    echo "[INFO] Restarting new master..."
    ssh   -i   $postgres_user_key   postgres@$new_master_name
"/etc/init.d/postgresql restart"
    status=$(ssh            -i            $postgres_user_key
postgres@$new_master_name "if ! killall -0 postgres; then
echo 'error'; else echo 'running'; fi;")
    if [ $status == "error" ]
    then
    echo "[ERROR] New postgres master not running !";
exit 0;
    else
        echo "[OK] New master started.";
    fi
    # Start new slave/master with online recovery
    echo "[INFO] Performing online slave recovery..."
    ssh -i $postgres_user_key postgres@$new_slave_name "bash
/var/lib/postgresql/streaming-replication.sh
$new_master_name"
    echo "[OK] Online recovery completed."
#Write changes to pgpool.conf file to keep the same current
master and slave nodes even after pgpool reboot.
    sed                                                  -i
"s/^backend_hostname0.*/backend_hostname0='$new_master_name'
/" $pgpool_configdir/pgpool.conf
    sed                                                  -i
"s/^backend_hostname1.*/backend_hostname1='$new_slave_name'/
" $pgpool_configdir/pgpool.conf

if [ $current_master_name == "192.167.70.x" ]
    then
      sed          -i          's/failover.sh/failover2.sh/g'
$pgpool_configdir/pgpool.conf
    fi

    if [ $current_master_name == "192.167.70.x" ]
    then
      sed          -i          's/failover2.sh/failover.sh/g'
$pgpool_configdir/pgpool.conf
    fi
```

```
echo "[OK] Pgpool configuration file updated."

    #Attach new master to pgpool
    echo "[INFO] Attaching new master node
'$new_master_name'..."
    AttachNodeToPgpool "$new_master_id"
    echo "[OK] New master node '$new_master_name' has been
successfully reattached to pgpool."

    #Attach new slave to pgpool
    echo "[INFO] Attaching new slave node
'$new_slave_name'..."
    AttachNodeToPgpool "$new_slave_id"
    echo "[OK] New slave node '$new_slave_name' has been
successfully reattached to pgpool."

    echo "[OK] PGPOOL RESTART."
    /etc/init.d/pgpool2 restart
    echo "[OK] END MASTER NODE REC."

}

CheckIfPgpoolIsRunning

#Get master/slave state
current_master_state=$(pcp_node_info 10 localhost 9898
pgpool pgpool $current_master_id | cut -d' ' -f3)
current_slave_state=$(pcp_node_info 10 localhost 9898 pgpool
pgpool $current_slave_id | cut -d' ' -f3)

# state 1 => postgres server is attached but still not
receiving connections
# state 2 => postgres server is attached and managing
clients connections
# state 3 => postgres server is detached and probably is
down.

#If slave is down and master is up then perform an online
slave backup.
[ $current_slave_state == 3 ] && ([ $current_master_state ==
1 ] || [ $current_master_state == 2 ]) &&
ReattachDegeneratedSlave
#If master is down then switch roles between failed
master(new server) and the slave(new master).
[ $current_master_state == 3 ] && SwitchOldMasterToSlave
```

**e. Configuración de Heartbeat.**

Heartbeat permite tener una alta disponiblidad de servicios en este caso del middleware, para esto de deberá tener tres archivos de configuración(en cada nodo) luego de la instalación del heartbeat `apt-get install heartbeat`.

La configuración de los archivos se realizara en la ruta /etc/ha.d

- ✓ ha.cf: fichero de configuración principal:

```
#archivo donde escribir los logs
debugfile /var/log/ha-debug

# archivo a donde escribir los mensajes
logfile/var/log/ha-log

#Facility to use for syslog()/logger
logfacilitylocal0

#keepalive: que tanto tiempo se debe esperar entre latidos.
keepalive 2

# deadtime: que tanto se debe esperar para declarar un host
muerto.
deadtime 30

# que tanto antes de que se lance una advertenia de un
"latido de corazón, retrazado" warntime 10
warntime 10
# Cuando se reinicia un sist. operativo puede haber un
retraso en el inicio del correcto funcionamiento de la red.

initdead 120

#initdead 100 What UDP port to use for bcast/ucast
communication. Qué puerto UDP usar para la comunicación
broadcast/unicast.
Udpport 694
```

```
#La frecuentia, baudios para puertos seriales

baud19200

#What interfaces to broadcast heartbeats over. En que
interfaz se debe lanzar el broadcast (para la comunicación
entre los modos.

bcasteth0

# Configurar un medio multicast para heartbeat

mcast eth0 225.0.0.1 694 1 0

# configurar un medio unicast/udp para heartbeat

ucast eth0 192.168.5.19

# Le decimos que cuando se restablesca el nodo maestro, le
retorne el control del servicio

auto_failback off

# Le decimos que máquinas están en el custer

node linuxha1

node linuxha2

# Agrega un pseudo miembro del cluster para que sea usado
por todos como referencia por si la conexión de red falla.
Idealmente debe ser un switch o un cluster que no forma
parte del custer. Es decir, van a estar haciendo ping a este
nodo, si no alcanzan respuesta dirán, ho no!!!, estamos
desconectados.
ping 192.168.5.1
```

- ✓ haresources: fichero de configuración de recursos:

```
mssp1 192.167.70.13 pgpool2 apache2
```

- ✓ authkeys: información de autenticación:

```
auth 1

1 sha1 passwd
```

### 3.2. Análisis, tratamiento de datos y presentación de resultados.

El Clúster de alta disponibilidad fue implementado en la empresa MINSE SENSE SOLUTIONS, siendo un total de 5 personas encargadas de los servidores y base de datos. Para el análisis y tratamiento de la información necesaria para este apartado si aplico un cuestionario del tipo Likert (Anexo 1) para medir el nivel de rendimiento del clúster y un segundo cuestionario para medir el nivel de escalabilidad en el diseño del clúster (Anexo 02).

Para medir el indicador de balance de carga se realizó una medición de una semana en las horas con mayor tráfico de datos (Anexo 03), para este esta medición se ha utilizado la herramienta System Monitor del Sistema Operativo Ubuntu Server 12.04 LTS la cual mide el tráfico de datos y el rendimiento del CPU en el sistema. Adicionalmente se ha utilizado una ficha de observación (Anexo 04) para medir el tiempo de inoperatividad del clúster ante la inclusión de un nodo y una encuesta (Anexo 05) para saber el tiempo estimado de inoperatividad para interrupciones previstas e imprevistas.

Los cuestionarios, fichas de información, mediciones de tráfico de datos y rendimiento de CPU, fueron aplicados antes y después del diseño e implementación del clúster de alta disponibilidad con el afán de que nos pueda permitir tener una perspectiva de la efectividad de la solución propuesta.

Los cuestionarios aplicados cuentan con seis y ocho preguntas respectivamente y para el procesamiento de los datos recogidos por estos, tanto para el nivel de rendimiento (tabla 6) y nivel de escalabilidad (tabla 9) se usaron el número de frecuencias de cada respuesta por cada encuestado, luego se obtuvo el total por respuesta (Muy Malo, Malo, Regular, Muy Bueno, Excelente).

A continuación se muestran los resultados obtenidos antes (Pre Test) y después (Post Test) del diseño e implementación del clúster.

### 3.2.1. Pre Test.

✓ **Indicador: Alto Rendimiento.**

Este indicador está referido a la capacidad de resistencia ante caídas, eficiencia energética y económica, rendimiento óptimo ante un número alto de transacciones, transferencia de información de manera eficiente e ininterrumpida.

Los resultados obtenidos fueron los siguientes:

| | Cantidad de Frecuencia | PORCENTAJE(%) |
|---|---|---|
| EXCELENTE | 0 | 0% |
| MUY BUENO | 0 | 0% |
| REGULAR | 0 | 0% |
| MALO | 9 | 30% |
| MUY MALO | 21 | 70% |
| **TOTAL** | **30** | **100%** |

Tabla 6. Resumen de nivel de rendimiento.

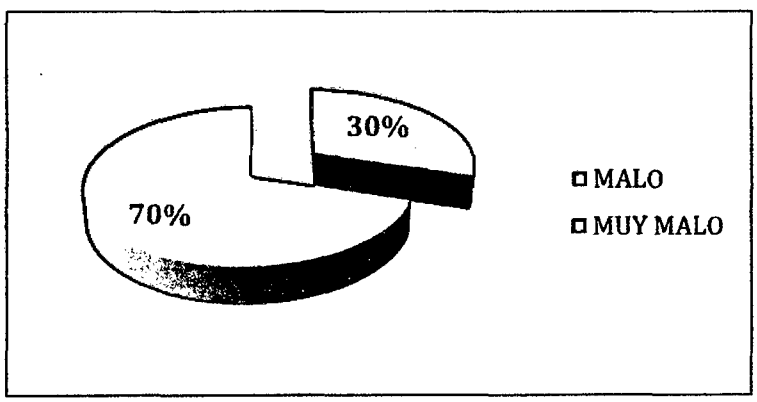



Gráfico 1. Nivel de rendimiento.

Los resultados obtenidos en el grafico anterior (Grafico 1) nos presenta el nivel de **rendimiento** antes del diseño e implementación de clúster, con los siguientes resultados: Muy Malo en 70% y Malo en 30%.

Los resultados obtenidos para la medición del **rendimiento** antes del diseño e implementación del clúster se obtuvieron de los siguientes criterios. Para ver el cuestionario completo realizado ver Anexo 1.

| ITEM | CRITERIOS |
|---|---|
| A | La arquitectura de servidores actual tiene resistencia a caídas (en cuanto cae un nodo, el otro se encarga de todos los servicios). |
| B | La arquitectura de servidores actual tiene eficiencia energética y económica. (migrar los equipos en caliente a un nodo y apagar otro por razones energéticas o de mantenimiento). |
| C | En la arquitectura de servidores actual, ante la caída de alguno de los ordenadores del clúster el servicio se puede ver mermado, pero mientras haya ordenadores en funcionamiento, éstos seguirán dando servicio. |
| D | En la arquitectura de servidores actual tiene la capacidad para hacer frente a volúmenes de trabajo cada vez mayores, prestando así un nivel de rendimiento óptimo. |
| E | La arquitectura de servidores actual puede transferir información y todo tipo de servicio por la red de forma rápida e ininterrumpidamente. |
| F | La arquitectura de servidores actual puede mantener estable la velocidad de respuesta al incrementar el número de transacciones. |

Tabla 7. Criterios de Cuestionario 01.

- ✓ **Indicador: Balanceo de Carga.**

Este indicador está referido a la técnica usada para compartir el trabajo a realizar entre varios procesos, ordenadores, discos u otros recursos. Está íntimamente ligado a los sistemas de multiprocesamiento, o que hacen uso de más de una unidad de procesamiento para realizar labores útiles

Los resultados obtenidos fueron los siguientes:

| | PROMEDIO POR HORAS | | | | | | |
|---|---|---|---|---|---|---|---|
| | HORA 01 | HORA 02 | HORA 03 | HORA 04 | HORA 05 | HORA 06 | HORA 07 |
| TRAFICO DE DATOS(KiB/s) | 45 | 40 | 35 | 60 | 80 | 70 | 45 |
| RENDIMIENTO CPU(%) | 70 | 60 | 55 | 60 | 70 | 55 | 35 |

Tabla 8. Resumen de Balanceo de Carga.

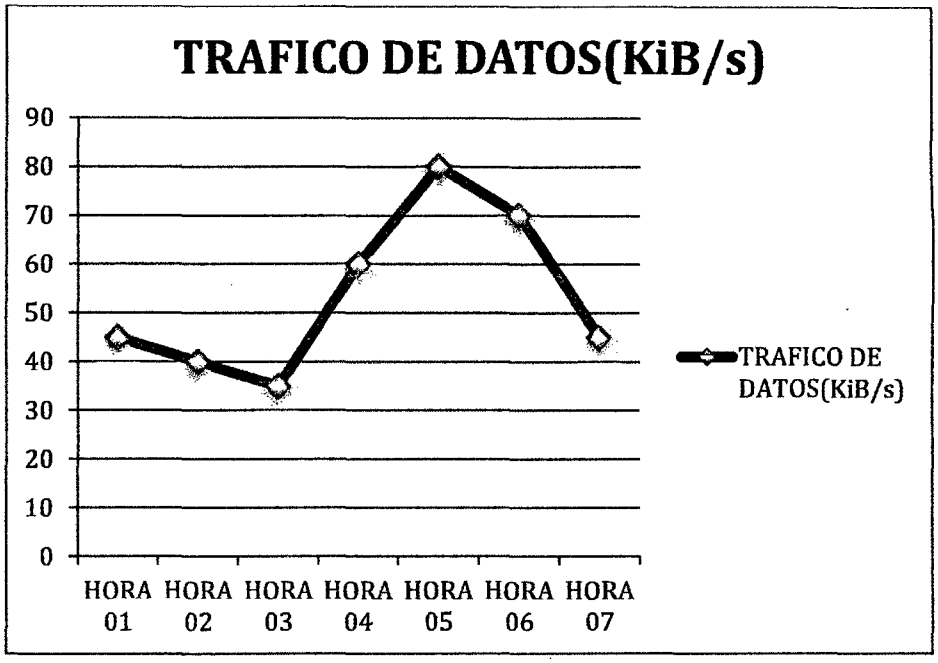



Gráfico 2. Tráfico de Datos.

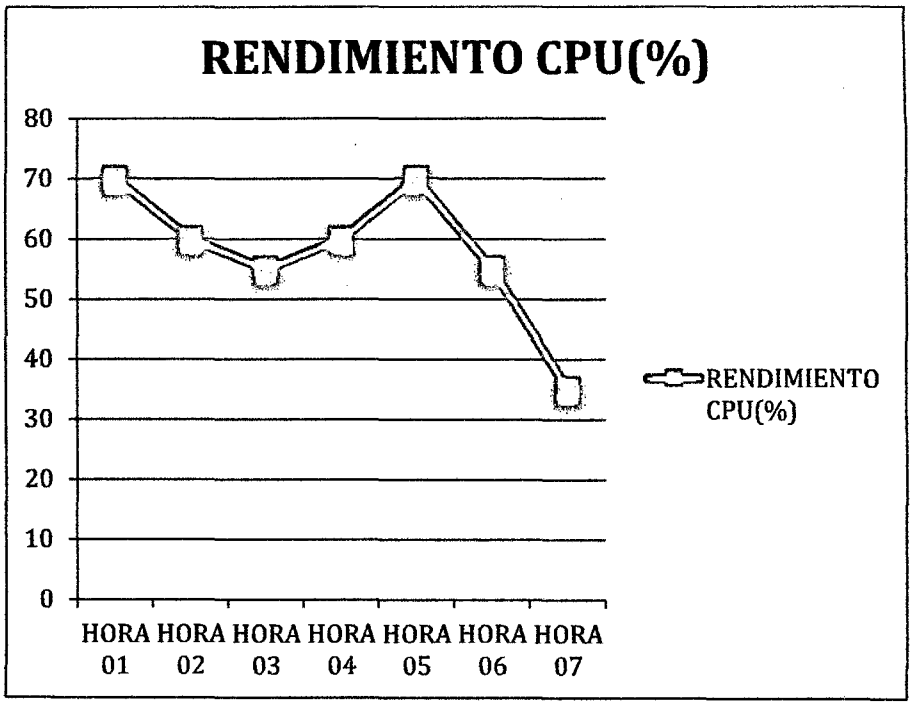



Gráfico 3. Rendimiento CPU.

Los resultados obtenidos en los gráficos anteriores (Grafico 2, Grafico 3) antes del diseño e implementación de clúster, obtuvieron los siguientes resultados: Tráfico de datos entre 35 KiB/S y 80 KiB/S, Rendimiento de CPU entre 35 % y 70%.

Los resultados obtenidos para la medición del **balanceo de cargo** antes del diseño e implementación del clúster se obtuvieron con mediciones durante una hora diaria en una semana con el sistema de monitoreo System Monitor del SO Ubuntu. Para ver las mediciones realizadas ver Anexo 2.

- ✓ **Indicador: Escalabilidad.**

Este indicador está referido a la propiedad deseable de un sistema, una red o un proceso, que indica su habilidad para reaccionar y adaptarse sin perder calidad, o bien manejar el crecimiento continuo de trabajo de manera fluida, o bien para estar preparado para hacerse más grande sin perder calidad en los servicios ofrecidos.

Los resultados obtenidos fueron los siguientes:

| | Cantidad de Frecuencia | PORCENTAJE(%) |
|---|---|---|
| EXCELENTE | 0 | 0% |
| MUY BUENO | 0 | 0% |
| REGULAR | 0 | 0% |
| MALO | 14 | 35% |
| MUY MALO | 26 | 65% |
| **TOTAL** | **40** | **100%** |

Tabla 9. Resumen de nivel de escalabilidad.

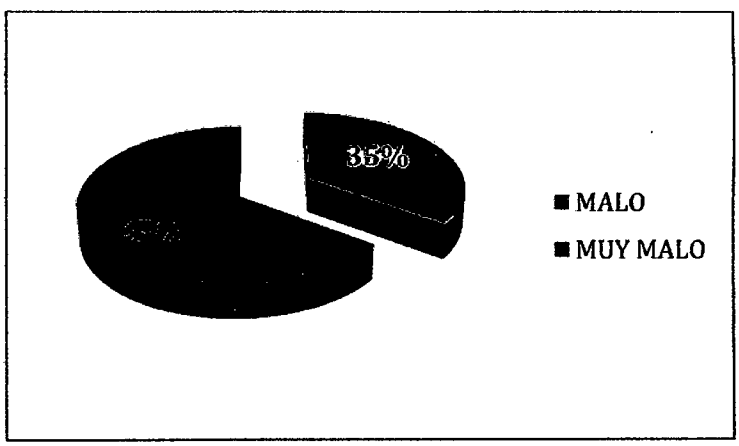



Gráfico 4. Nivel de escalabilidad.

Los resultados obtenidos en el grafico anterior (Grafico 04) nos presenta el nivel de **escalabilidad** antes del diseño e implementación de clúster, con los siguientes resultados: Muy Malo en 65% y Malo en 35%.

Los resultados obtenidos para la medición de la **escalabilidad** antes del diseño e implementación del clúster se obtuvieron de los siguientes criterios. Para ver el cuestionario completo realizado ver Anexo 3.

| | |
|---|---|
| A | La arquitectura de servidores actual está preparada para hacerse más grande sin perder calidad en los servicios ofrecidos. |
| B | La arquitectura de servidores actual tiene la capacidad de cambiar su tamaño o configuración para adaptarse a las circunstancias cambiantes. |
| C | La arquitectura de servidores actual puede acomodar a un número creciente de usuarios mediante la adición de hardware, sin tener que modificar el software. |
| D | La arquitectura de servidores actual tiene la capacidad de ampliar y reducir sus recursos para acomodar (a conveniencia), cargas más pesadas o más ligeras según se requiera. |
| E | La arquitectura de servidores actual puede ser utilizada para poder procesar más transacciones añadiendo por medio de nuevos procesadores, dispositivos y almacenamiento que se pueden implementar fácil y transparentemente sin apagarlos. |
| F | La arquitectura de servidores actual tiene la habilidad de reaccionar y adaptarse al crecimiento continuo de trabajo de manera fluida. |
| G | La arquitectura de servidores actual puede ampliar su capacidad fácilmente añadiendo más ordenadores al clúster. |
| H | En la arquitectura de servidores actual tiene la capacidad de aumentar sus capacidades a través de mejores técnicas. |

Tabla 10. Criterios de Cuestionario 02.

- ✓ **Indicador: Indicador de Alta Disponibilidad.**

Este indicador está referido al porcentaje del tiempo de funcionamiento en un año dado.

Los resultados obtenidos fueron los siguientes:

| | Cantidad(al año) | Tiempo Estimado (minutos) | TOTAL NO DISPONIBLE (minutos) |
|---|---|---|---|
| Interrupciones Previstas | 3 | 180 | 540 |
| Interrupciones Imprevistas | 2 | 240 | 480 |
| | | TOTAL | 1020 |

Tabla 11. Interrupciones previstas e imprevistas.

| | | |
|---|---|---|
| | | TOTAL NO DISPONIBLE (minutos) |
| Interrupciones Previstas | | 540 |
| Interrupciones Imprevistas | | 480 |

Tabla 12. Resumen Interrupciones previstas e imprevistas.

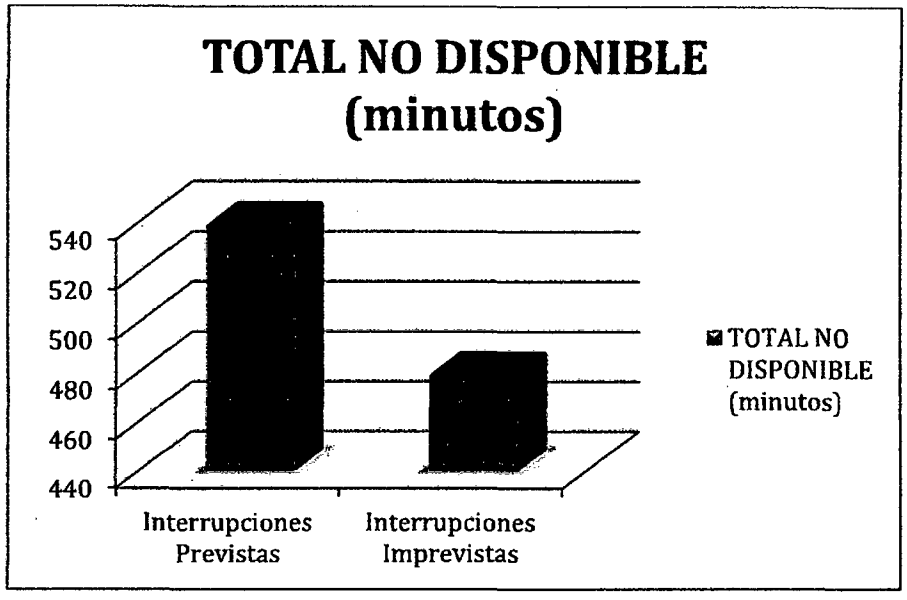



Gráfico 5. Interrupciones previstas e imprevistas.

Los resultados obtenidos en el gráfico anterior (Grafico 5) antes del diseño e implementación de clúster, muestra siguientes resultados: Interrupciones Previstas 540 minutos e Interrupciones Imprevistas 480 minutos.

Los resultados obtenidos para la medición del **indicador de alta disponibilidad** antes del diseño e implementación del clúster se obtuvieron con una encuesta de estimación de interrupciones en el servicio durante un año. Para ver el cuestionario completo realizado ver Anexo 5.

### 3.2.2. Post Test.

✓ **Indicador: Alto Rendimiento.**

Los resultados obtenidos fueron los siguientes:

| | Cantidad de Frecuencia | PORCENTAJE(%) |
|---|---|---|
| EXCELENTE | 18 | 60% |
| MUY BUENO | 12 | 40% |
| REGULAR | 0 | 0% |
| MALO | 0 | 0% |
| MUY MALO | 0 | 0% |
| **TOTAL** | **30** | **100%** |

Tabla 13. Resumen de nivel de rendimiento.

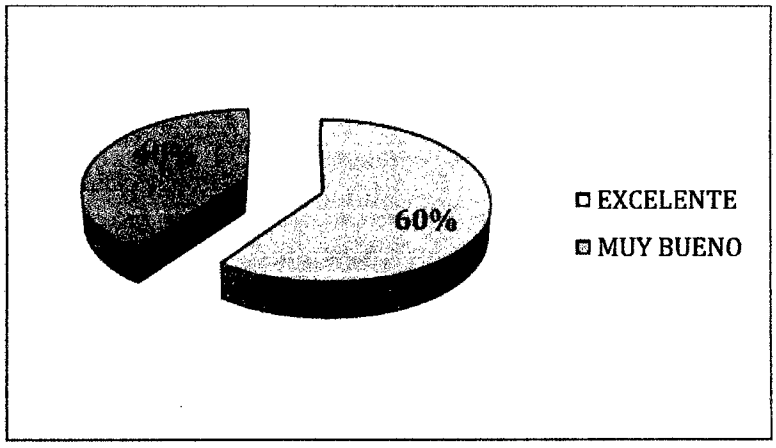



Gráfico 6. Nivel de rendimiento.

Los resultados obtenidos en el grafico anterior (Grafico 6) nos presenta el nivel de **rendimiento** después del diseño e implementación de clúster, con los siguientes resultados: Muy Bueno en 40% y Excelente en 60%.

Los resultados obtenidos para la medición del **rendimiento** después del diseño e implementación del clúster se obtuvieron de la misma encuesta aplicada para el antes.

✓ **Indicador: Balanceo de Carga.**

Los resultados obtenidos fueron los siguientes:

| | PROMEDIO POR HORAS | | | | | | |
|---|---|---|---|---|---|---|---|
| | HORA 01 | HORA 02 | HORA 03 | HORA 04 | HORA 05 | HORA 06 | HORA 07 |
| TRAFICO DE DATOS(KiB/s) | 15 | 15 | 25 | 12 | 20 | 5 | 7 |
| RENDIMIENTO CPU(%) | 10 | 11 | 25 | 15 | 30 | 8 | 6 |

Tabla 14. Resumen de Balanceo de Carga.

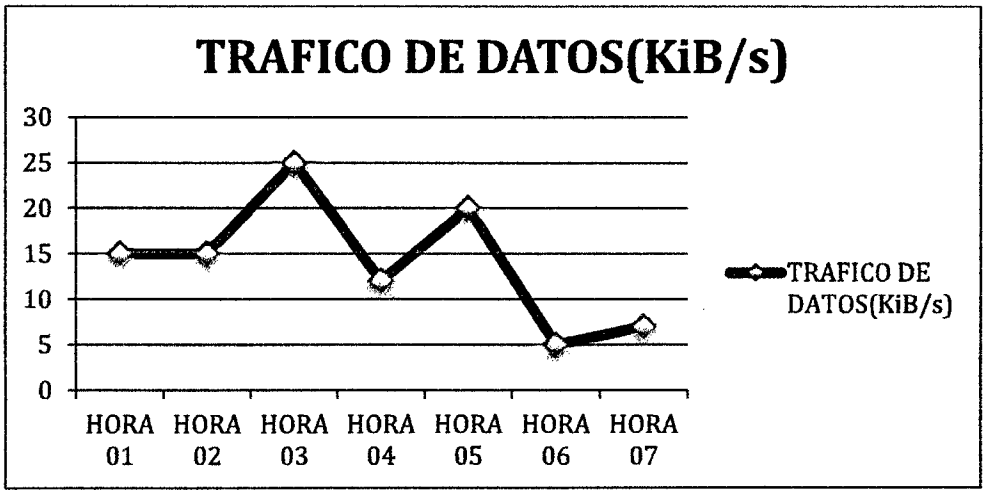



Gráfico 7. Tráfico de datos.

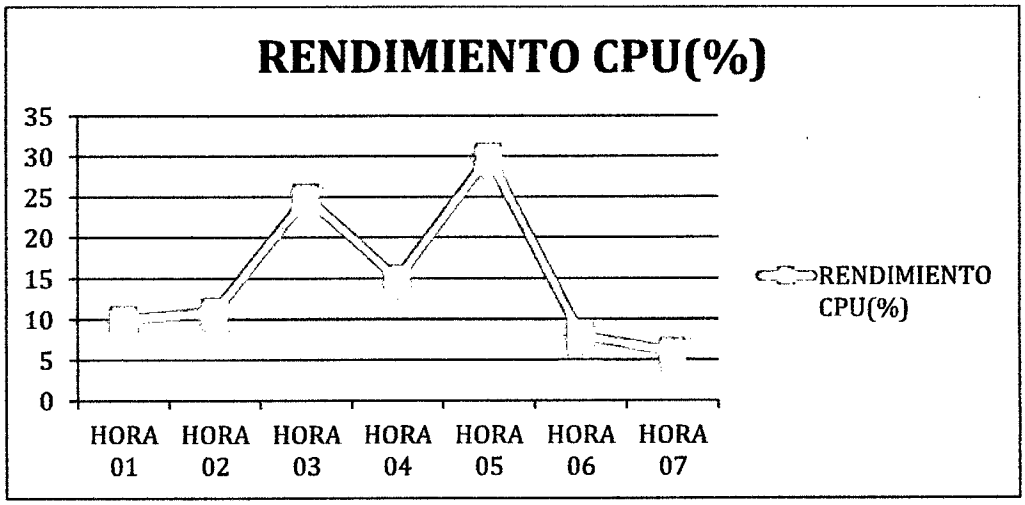



Gráfico 8. Rendimiento de CPU.

Los resultados obtenidos en los gráficos anteriores (Grafico 7, Grafico 8) después del diseño e implementación de clúster, obtuvieron los siguientes resultados: Tráfico de datos entre 5 KiB/S y 25 KiB/S, Rendimiento de CPU entre 5 % y 30%.

Los resultados obtenidos para la medición del balanceo de cargo después del diseño e implementación del clúster se obtuvieron con mediciones durante una hora diaria en una semana con el sistema de monitoreo System Monitor del SO Ubuntu.

✓ **Indicador: Escalabilidad.**

Los resultados obtenidos fueron los siguientes:

| | Cantidad de Frecuencia | PORCENTAJE(%) |
|---|---|---|
| EXCELENTE | 29 | 72% |
| MUY BUENO | 11 | 28% |
| REGULAR | 0 | 0% |
| MALO | 0 | 0% |
| MUY MALO | 0 | 0% |
| **TOTAL** | **40** | **100%** |

Tabla 15. Resumen de nivel de escalabilidad.

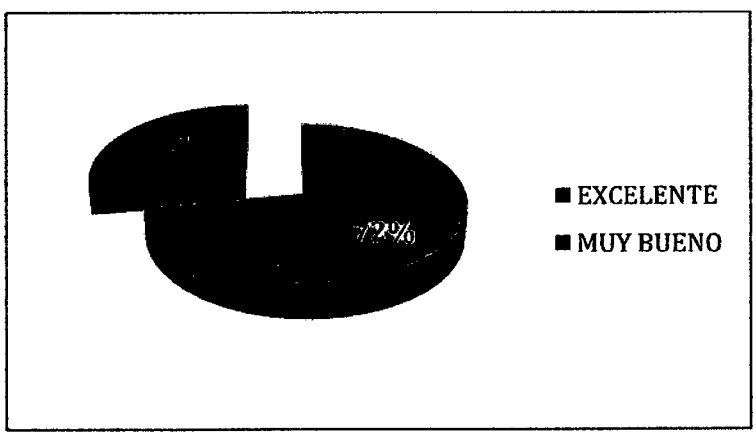



Gráfico 9. Nivel de Escalabilidad.

Los resultados obtenidos en el grafico anterior (Grafico 9) nos presenta el nivel de **escalabilidad** después del diseño e implementación de clúster, con los siguientes resultados: Muy Bueno en 28% y Excelente en 72%.

Los resultados obtenidos para la medición de la **escalabilidad** después del diseño e implementación del clúster se obtuvieron de la misma encuesta aplicada para el antes.

- ✓ **Indicador: Indicador de Alta Disponibilidad.**

Los resultados obtenidos fueron los siguientes:

| | Cantidad(al año) | Tiempo Estimado (minutos) | TOTAL NO DISPONIBLE (minutos) |
|---|---|---|---|
| Interrupciones Previstas | 3 | 5 | 15 |
| Interrupciones Imprevistas | 2 | 5 | 10 |
| | | TOTAL | 25 |

Tabla 16. Interrupciones previstas e imprevistas.

| | TOTAL NO DISPONIBLE (minutos) |
|---|---|
| Interrupciones Previstas | 15 |
| Interrupciones Imprevistas | 10 |

Tabla 17. Resumen Interrupciones previstas e imprevistas.

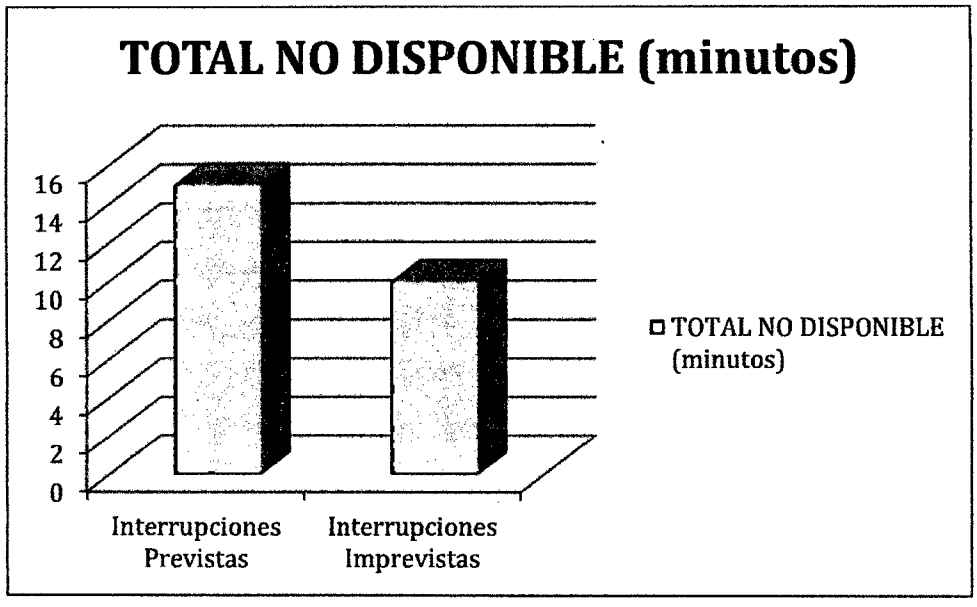



Gráfico 10. Interrupciones previstas e imprevistas.

Los resultados obtenidos en el gráfico anterior (Grafico 10) después del diseño e implementación de clúster, muestra siguientes resultados: Interrupciones Previstas 15 minutos e Interrupciones Imprevistas 10 minutos.

Los resultados obtenidos para la medición del **indicador de alta disponibilidad** antes del diseño e implementación del clúster se obtuvieron con una encuesta de estimación de interrupciones en el servicio durante un año y una ficha de observación (Anexo 4) del tiempo que tomar en restablecerse un nodo en el clúster.

## CAPITULO IV. ANÁLISIS Y DISCUSIÓN DE RESULTADOS

Los resultados obtenidos de las encuestas, fichas de observación y mediciones, mostradas en el capítulo anterior, se realizarán las pruebas de hipótesis correspondientes con el fin de analizar los indicadores de **alto rendimiento, balanceo de carga, escalabilidad e indicador de alta disponibilidad** los cuales medirán el efecto de la variable dependiente (**Alta disponibilidad de la información**) tras la manipulación de la variable independiente (**Implementación de un clúster de servidores Linux**).

### 4.1. Análisis de resultados.

#### 4.1.1. Prueba de hipótesis para el primer indicador: Alto Rendimiento

***a. Formulación de las hipótesis***

*Hipótesis Nula:*

**$H_0$:** La implementación de un clúster de alta disponibilidad no aumenta el nivel de rendimiento del servicio de acceso a información.

*Hipótesis Alterna:*

**$H_a$:** La implementación de un clúster de alta disponibilidad aumenta el nivel de rendimiento del servicio de acceso a información.

***b. Elección del nivel de significancia o confianza***

El nivel de significancia será del 5%, $\alpha = 0.05$.

***c. Elección del estadístico de prueba***

Por tener una muestra igual a la cantidad de la población que son 5 encuestados y al ser esta una muestra $n < 30$, se aplicará la prueba

estadística *t*-student para muestras emparejadas, utilizada para medir muestras medidas en más de un tiempo. Para este caso mediremos el nivel de rendimiento antes de la implementación del clúster (Pre-test) y el nivel de rendimiento después del diseño e implementación del clúster de alta disponibilidad (Post-test).

| NIVEL DE ALTO RENDIMIENTO | ANTES | DESPUES | DIFERENCIA |
|---|---|---|---|
| Entrevistado 1 | 7 | 27 | -20 |
| Entrevistado 2 | 8 | 26 | -18 |
| Entrevistado 3 | 8 | 27 | -19 |
| Entrevistado 4 | 9 | 28 | -19 |
| Entrevistado 5 | 7 | 30 | -23 |

Tabla 18. Análisis-Resultado para Alto Rendimiento.

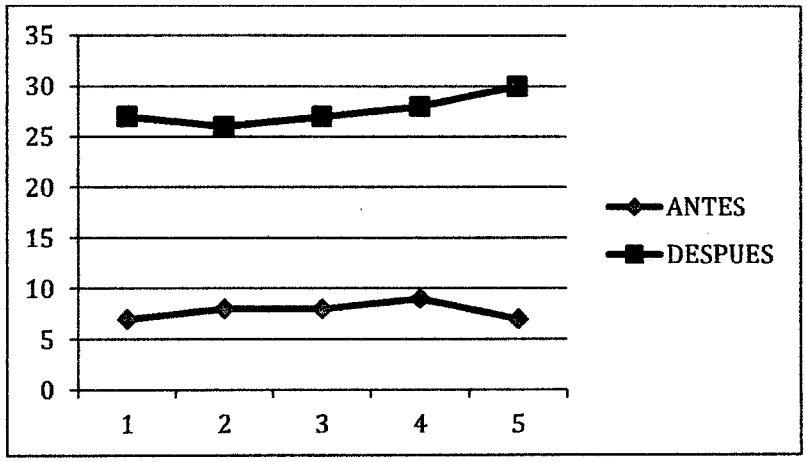



Gráfico 11. Nivel de Rendimiento (Antes, Después).

Luego se aplica un análisis de estadística descriptiva de la columna **DIFERENCIA** para hallar la ***Media*** y la ***Desviación Estándar.***

| DIFERENCIA | |
|---|---|
| | |
| Media | -19.8 |
| Error típico | 0.860232527 |
| Mediana | -19 |
| Moda | -19 |
| Desviación estándar | 1.923538406 |
| Varianza de la muestr | 3.7 |
| Curtosis | 2.607742878 |
| Coeficiente de asimet | -1.517473688 |
| Rango | 5 |
| Mínimo | -23 |
| Máximo | -18 |
| Suma | -99 |
| Cuenta | 5 |
| Nivel de confianza(95 | 2.388388388 |

Luego, para obtener el estadístico de prueba, se aplicará la siguiente fórmula.

$$t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}} = -23.01703$$

Cálculo estadístico de la prueba t para dos muestra emparejadas:

Prueba t para medias de dos muestras emparejadas

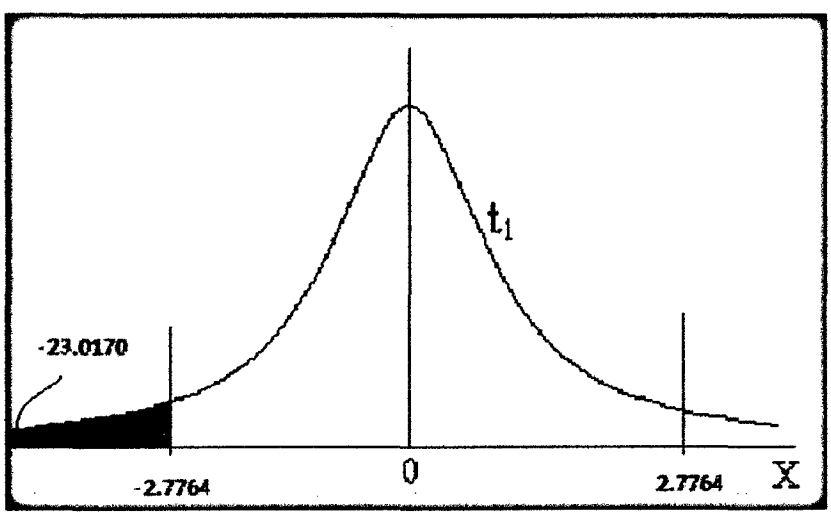
| | *ANTES* | *DESPUES* |
|---|---|---|
| Media | 7.8 | 27.6 |
| Varianza | 0.7 | 2.3 |
| Observaciones | 5 | 5 |
| Coeficiente de correlación de Pearson | -0.27583864 | |
| Diferencia hipotética de las medias | 0 | |
| Grados de libertad | 4 | |
| Estadístico t | -23.0170325 | |
| P(T<=t) una cola | 0.00001056 | |
| Valor crítico de t (una cola) | 2.131846786 | |
| P(T<=t) dos colas | 0.00002111 | |
| Valor crítico de t (dos colas) | 2.776445105 | |

### *d. Cálculo del valor crítico de la estadística de prueba*

De acuerdo a los datos obtenidos, aplicando la prueba *t* para dos medidas de muestras emparejadas y usando $\alpha = 0.05$, tenemos como valor crítico:

$$- t_{0.0000337} \leq -2.7764 \text{ y } t_{0.0000337} \geq 2.7764$$

### *e. Definición de la regla de decisión*



| | |
|---|---|
| **Hipótesis** | $H_0$: μd <= 0<br>$H_a$: μd > 0 |
| **Estadístico de prueba** | $t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}}$ |
| **Regla de rechazo:**<br>**Método *p* - value** | Rechazar $H_0$ si *p* - value ≤ α |
| **Regla de rechazo:**<br>**Método del valor crítico** | Rechazar $H_0$ si $t \leq$ - 2.7764 o<br>Rechazar $H_0$ si $t \geq$ 2.7764 |

***f. Toma de decisión de aceptar o rechazar Ho***

Como el *p* - value = 0.0000211 es menor que el nivel de significancia α = 0.05, y como el valor del estadístico de prueba *t* = -23.0170325 es menor que el valor crítico = - 2.7764. Se tiene evidencia suficiente para rechazar la hipótesis nula.

Por lo tanto, se acepta la hipótesis alterna $H_a$: **La implementación de un clúster de alta disponibilidad aumenta el nivel de rendimiento del servicio de acceso a información.**

### 4.1.2. Prueba de hipótesis para el segundo indicador: Balanceo de Carga.

***a. Formulación de las hipótesis***

*Hipótesis Nula:*

**$H_0$:** La implementación de un clúster de alta disponibilidad no disminuye la carga en el tráfico de datos (KiB/S) y en el rendimiento de CPU (%) en el servidor.

*Hipótesis Alterna:*

**$H_a$:** La implementación de un clúster de alta disponibilidad disminuye la carga en el tráfico de datos (KiB/S) y en el rendimiento de CPU (%) en el servidor.

### b. *Elección del nivel de significancia o confianza*

El nivel de significancia será del 5%, $\alpha = 0.05$.

### c. *Elección del estadístico de prueba*

Por tener una muestra igual a la cantidad de la población que son 5 encuestados y al ser esta una muestra $n < 30$, se aplicará la prueba estadística *t*-student para muestras emparejadas, utilizada para medir muestras medidas en más de un tiempo. Para este caso mediremos el nivel de rendimiento antes de la implementación del clúster (Pre-test) y el nivel de rendimiento después del diseño e implementación del clúster de alta disponibilidad (Post-test).

| NIVEL DE RENDIMIENTO CPU(%) | ANTES | DESPUES | DIFERENCIA |
|---|---|---|---|
| Hora 1 | 70 | 10 | 60 |
| Hora 2 | 60 | 11 | 49 |
| Hora 3 | 55 | 25 | 30 |
| Hora 4 | 60 | 15 | 45 |
| Hora 5 | 70 | 30 | 40 |
| Hora 6 | 55 | 8 | 47 |
| Hora 7 | 35 | 6 | 29 |

Tabla 19. Análisis-Resultado para Rendimiento de CPU.

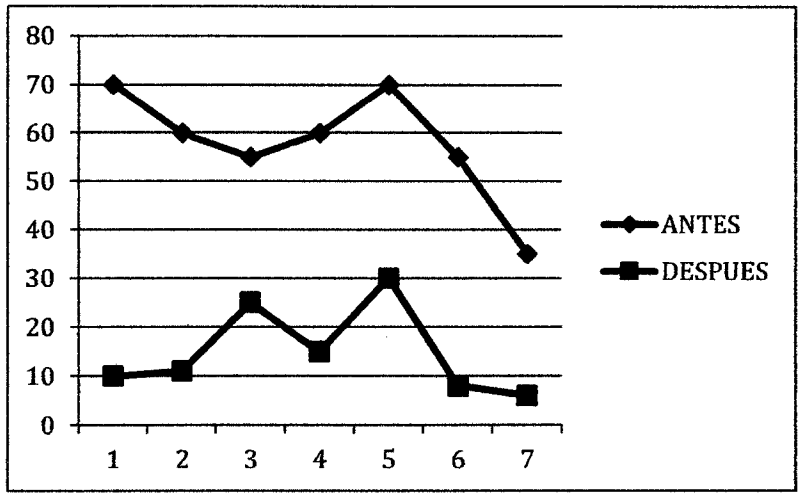



Gráfico 12. Rendimiento CPU (Antes, Después).

Luego se aplica un análisis de estadística descriptiva de la columna **DIFERENCIA** para hallar la ***Media*** y la ***Desviación Estándar.***

| DIFERENCIA | |
|---|---|
| | |
| Media | 42.8571429 |
| Error típico | 4.13710602 |
| Mediana | 45 |
| Moda | |
| Desviación estándar | 10.9457537 |
| Varianza de la muestra | 119.809524 |
| Curtosis | -0.4464293 |
| Coeficiente de asimetría | 0.0962762 |
| Rango | 31 |
| Mínimo | 29 |
| Máximo | 60 |
| Suma | 300 |
| Cuenta | 7 |
| Nivel de confianza(95.0%) | 10.1231338 |

Luego, para obtener el estadístico de prueba, se aplicará la siguiente fórmula.

$$t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}} = 10.35920825$$

Cálculo estadístico de la prueba t para dos muestra emparejadas:

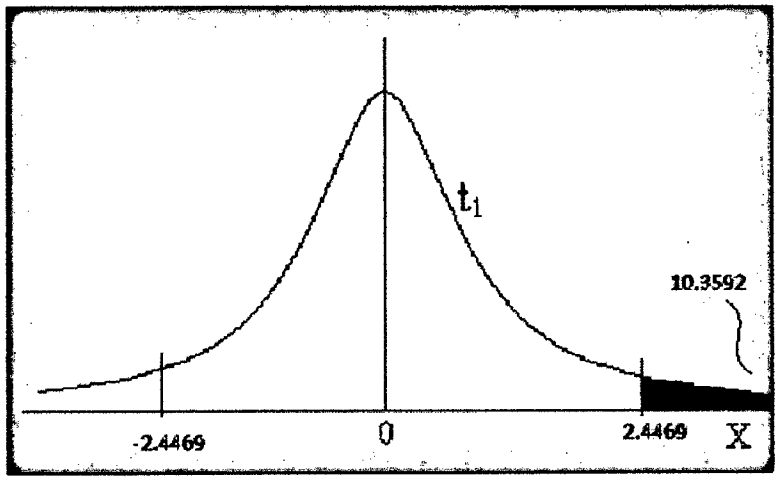
| Prueba t para medias de dos muestras emparejadas | | |
|---|---|---|
| | | |
| | *ANTES* | *DESPUES* |
| Media | 57.85714286 | 15 |
| Varianza | 140.4761905 | 82.66666667 |
| Observaciones | 7 | 7 |
| Coeficiente de correlación de Pearson | 0.479450604 | |
| Diferencia hipotética de las medias | 0 | |
| Grados de libertad | 6 | |
| Estadístico t | 10.35920825 | |
| P(T<=t) una cola | 0.000023673 | |
| Valor crítico de t (una cola) | 1.943180281 | |
| P(T<=t) dos colas | 0.000047346 | |
| Valor crítico de t (dos colas) | 2.446911851 | |

***d. Cálculo del valor crítico de la estadística de prueba***

De acuerdo a los datos obtenidos, aplicando la prueba *t* para dos medidas de muestras emparejadas y usando $\alpha = 0.05$, tenemos como valor crítico:

$$- t_{0.0000316} \leq -2.4469 \text{ y } t_{0.0000316} \geq 2.4469$$

***e. Definición de la regla de decisión***



| | |
|---|---|
| **Hipótesis** | $H_0$: μd <= 0<br>$H_a$: μd > 0 |
| **Estadístico de prueba** | $t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}}$ |
| **Regla de rechazo:**<br>**Método *p* - value** | Rechazar $H_0$ si *p* - value ≤ α |
| **Regla de rechazo:**<br>**Método del valor crítico** | Rechazar $H_0$ si $t \leq$ - 2.4469 o<br>Rechazar $H_0$ si $t \geq$ 2.4469 |

### *f. Toma de decisión de aceptar o rechazar Ho*

Como el *p* - value = 0.00003734 es menor que el nivel de significancia α = 0.05, y como el valor del estadístico de prueba *t* = 10.35920825 es mayor que el valor crítico = 2.4469. Se tiene evidencia suficiente para rechazar la hipótesis nula.

Por lo tanto, se acepta la hipótesis alterna $H_a$: **La implementación de un clúster de alta disponibilidad disminuye la carga en el rendimiento de CPU (%) del servidor.**

| NIVEL DE TRAFICO DE DATOS | ANTES | DESPUES | DIFERENCIA |
|---|---|---|---|
| Hora 1 | 45 | 15 | 30 |
| Hora 2 | 40 | 15 | 25 |
| Hora 3 | 35 | 25 | 10 |
| Hora 4 | 60 | 12 | 48 |
| Hora 5 | 80 | 20 | 60 |
| Hora 6 | 70 | 5 | 65 |
| Hora 7 | 45 | 7 | 38 |

Tabla 20. Análisis-Resultado para Tráfico de Datos.

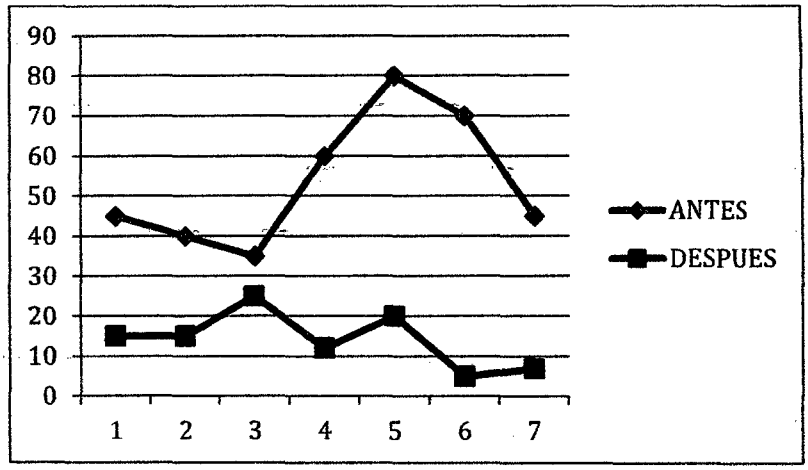



Gráfico 13. Tráfico de Datos (Antes, Después).

Luego se aplica un análisis de estadística descriptiva de la columna **DIFERENCIA** para hallar la ***Media*** y la ***Desviación Estándar.***

| *DIFERENCIA* | |
|---|---|
| | |
| Media | 39.4285714 |
| Error típico | 7.42536561 |
| Mediana | 38 |
| Moda | |
| Desviación estándar | 19.6456708 |
| Varianza de la muestra | 385.952381 |
| Curtosis | -0.9722241 |
| Coeficiente de asimetría | -0.1007799 |
| Rango | 55 |
| Mínimo | 10 |
| Máximo | 65 |
| Suma | 276 |
| Cuenta | 7 |
| Nivel de confianza(95.0%) | 18.1692151 |

Luego, para obtener el estadístico de prueba, se aplicará la siguiente fórmula.

$$t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}} = 5.309983$$

Cálculo estadístico de la prueba t para dos muestra emparejadas:

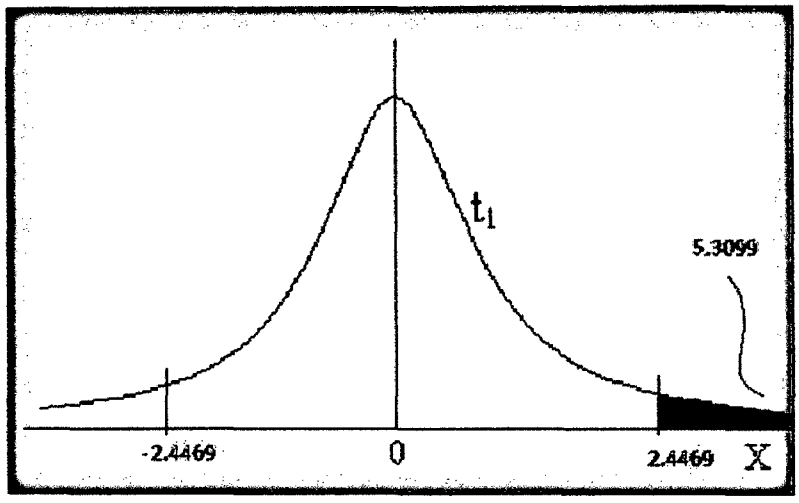
| Prueba t para medias de dos muestras emparejadas | | |
|---|---|---|
| | | |
| | *ANTES* | *DESPUES* |
| Media | 53.57142857 | 14.14285714 |
| Varianza | 280.952381 | 48.80952381 |
| Observaciones | 7 | 7 |
| Coeficiente de correlación de Pearson | -0.239918685 | |
| Diferencia hipotética de las medias | 0 | |
| Grados de libertad | 6 | |
| Estadístico t | 5.309983845 | |
| P(T<=t) una cola | 0.000906217 | |
| Valor crítico de t (una cola) | 1.943180281 | |
| P(T<=t) dos colas | 0.001812435 | |
| Valor crítico de t (dos colas) | 2.446911851 | |

### *g. Cálculo del valor crítico de la estadística de prueba*

De acuerdo a los datos obtenidos, aplicando la prueba *t* para dos medidas de muestras emparejadas y usando $\alpha = 0.05$, tenemos como valor crítico:

$$-t_{0.001812435} \leq -2.4469 \text{ y } t_{0.001812435} \geq 2.4469$$

### *h. Definición de la regla de decisión*




| | |
|---|---|
| **Hipótesis** | $H_0$: μd <= 0<br>$H_a$: μd > 0 |
| **Estadístico de prueba** | $t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}}$ |
| **Regla de rechazo:**<br>**Método *p* - value** | Rechazar $H_0$ si *p* - value ≤ α |
| **Regla de rechazo:**<br>**Método del valor crítico** | Rechazar $H_0$ si $t \leq$ - 2.4469 o<br>Rechazar $H_0$ si $t \geq$ 2.4469 |

### *i. Toma de decisión de aceptar o rechazar Ho*

Como el *p* - value = 0.001812435 es menor que el nivel de significancia α = 0.05, y como el valor del estadístico de prueba *t* = 5.309983845 es mayor que el valor crítico = 2.4469. Se tiene evidencia suficiente para rechazar la hipótesis nula.

Por lo tanto, se acepta la hipótesis alterna $H_a$: **La implementación de un clúster de alta disponibilidad disminuye la carga en el tráfico de datos (KiB/S).**

### 4.1.3. Prueba de hipótesis para el tercer indicador: Escalabilidad.

***a. Formulación de las hipótesis***

*Hipótesis Nula:*

**$H_0$:** La implementación de un clúster de alta disponibilidad no aumenta el nivel de escalabilidad del servicio de acceso a información.

*Hipótesis Alterna:*

**$H_a$:** La implementación de un clúster de alta disponibilidad aumenta el nivel de escalabilidad del servicio de acceso a información.

***b. Elección del nivel de significancia o confianza***

El nivel de significancia será del 5%, $\alpha = 0.05$.

***c. Elección del estadístico de prueba***

Por tener una muestra igual a la cantidad de la población que son 5 encuestados y al ser esta una muestra $n < 30$, se aplicará la prueba estadística *t*-student para muestras emparejadas, utilizada para medir muestras medidas en más de un tiempo. Para este caso mediremos el nivel de rendimiento antes de la implementación del clúster (Pre-test) y el nivel de rendimiento después del diseño e implementación del clúster de alta disponibilidad (Post-test).

| NIVEL DE ESCALABILIDAD | ANTES | DESPUES | DIFERENCIA |
|---|---|---|---|
| Entrevistado 1 | 11 | 37 | -26 |
| Entrevistado 2 | 12 | 38 | -26 |
| Entrevistado 3 | 10 | 38 | -28 |
| Entrevistado 4 | 10 | 39 | -29 |
| Entrevistado 5 | 11 | 37 | -26 |

Tabla 21. Análisis-Resultado para nivel de Escalabilidad.

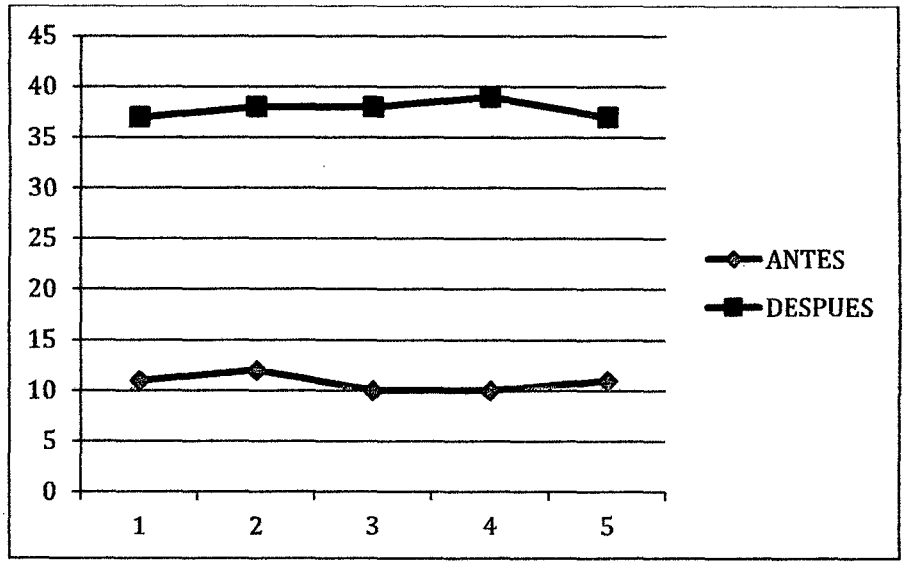



Gráfico 14. Nivel de Escalabilidad (Antes, Después).

Luego se aplica un análisis de estadística descriptiva de la columna **DIFERENCIA** para hallar la ***Media*** y la ***Desviación Estándar.***

| *DIFERENCIA* | |
|---|---|
| | |
| Media | -27 |
| Error típico | 0.632455532 |
| Mediana | -26 |
| Moda | -26 |
| Desviación estándar | 1.414213562 |
| Varianza de la muestra | 2 |
| Curtosis | -1.75 |
| Coeficiente de asimetría | -0.883883476 |
| Rango | 3 |
| Mínimo | -29 |
| Máximo | -26 |
| Suma | -135 |
| Cuenta | 5 |
| Nivel de confianza(95.0%) | 1.755978066 |

Luego, para obtener el estadístico de prueba, se aplicará la siguiente fórmula.

$$t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}} = -42.690748$$

Cálculo estadístico de la prueba t para dos muestra emparejadas:

Prueba t para medias de dos muestras emparejadas

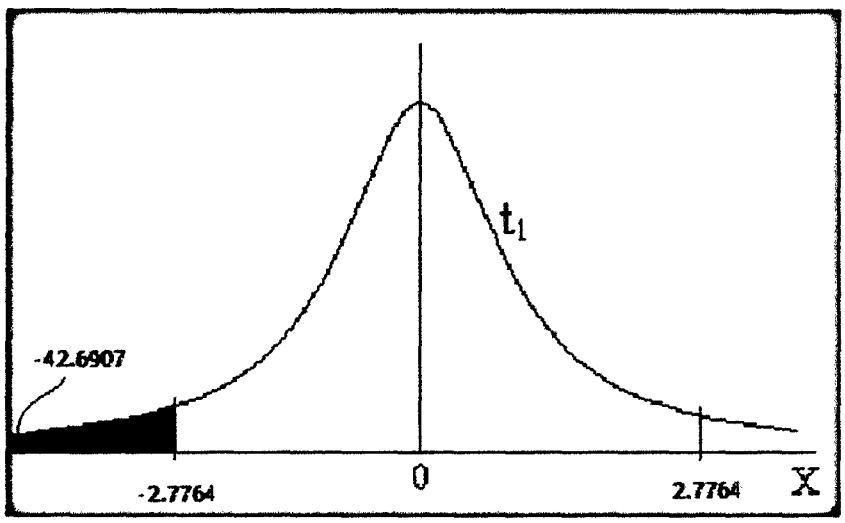
| | *ANTES* | *DESPUES* |
|---|---|---|
| Media | 10.8 | 37.8 |
| Varianza | 0.7 | 0.7 |
| Observaciones | 5 | 5 |
| Coeficiente de correlación de Pearson | -0.428571429 | |
| Diferencia hipotética de las medias | 0 | |
| Grados de libertad | 4 | |
| Estadístico t | -42.69074841 | |
| P(T<=t) una cola | 0.00000090 | |
| Valor crítico de t (una cola) | 2.131846786 | |
| P(T<=t) dos colas | 0.000001800 | |
| Valor crítico de t (dos colas) | 2.776445105 | |

***d. Cálculo del valor crítico de la estadística de prueba***

De acuerdo a los datos obtenidos, aplicando la prueba *t* para dos medidas de muestras emparejadas y usando $\alpha = 0.05$, tenemos como valor crítico:

$$- t_{0.000001013} \leq -2.7764 \text{ y } t_{0.000001013} \geq 2.7764$$

***e. Definición de la regla de decisión***



| | |
|---|---|
| **Hipótesis** | $H_0$: μd <= 0<br>$H_a$: μd > 0 |
| **Estadístico de prueba** | $t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}}$ |
| **Regla de rechazo:**<br>**Método *p* - value** | Rechazar $H_0$ si *p* - value ≤ α |
| **Regla de rechazo:**<br>**Método del valor crítico** | Rechazar $H_0$ si $t \leq$ - 2.7764 o<br>Rechazar $H_0$ si $t \geq$ 2.7764 |

***f. Toma de decisión de aceptar o rechazar Ho***

Como el *p* - value = 0.000001800 es menor que el nivel de significancia α = 0.05, y como el valor del estadístico de prueba *t* = -42.690748 es menor que el valor crítico = - 2.7764. Se tiene evidencia suficiente para rechazar la hipótesis nula.

Por lo tanto, se acepta la hipótesis alterna $H_a$: **La implementación de un clúster de alta disponibilidad aumenta el nivel de escalabilidad del servicio de acceso a información.**

### 4.1.4. Prueba de hipótesis para el tercer indicador: Alta Disponibilidad.

***a. Formulación de las hipótesis***

*Hipótesis Nula:*

**$H_0$:** La implementación de un clúster de alta disponibilidad no disminuye el tiempo de inactividad del servicio de acceso a información.

*Hipótesis Alterna:*

**$H_a$:** La implementación de un clúster de alta disponibilidad disminuye el tiempo de inactividad del servicio de acceso a información.

### *b. Elección del nivel de significancia o confianza*

El nivel de significancia será del 5%, $\alpha = 0.05$.

### *c. Elección del estadístico de prueba*

Por tener una muestra igual a la cantidad de la población que son 5 encuestados y al ser esta una muestra $n < 30$, se aplicará la prueba estadística t-student para muestras emparejadas, utilizada para medir muestras medidas en más de un tiempo. Para este caso mediremos el nivel de rendimiento antes de la implementación del clúster (Pre-test) y el nivel de rendimiento después del diseño e implementación del clúster de alta disponibilidad (Post-test).

| | ANTES | DESPUES | DIFERENCIA | Tiempo Reducido |
|---|---|---|---|---|
| Interrupciones Previstas (min) | 540 | 15 | 525 | **97.22%** |
| Interrupciones Imprevistas (min) | 480 | 10 | 470 | **97.92%** |

Tabla 22. Análisis-Resultado para Interrupciones previstas e imprevistas.

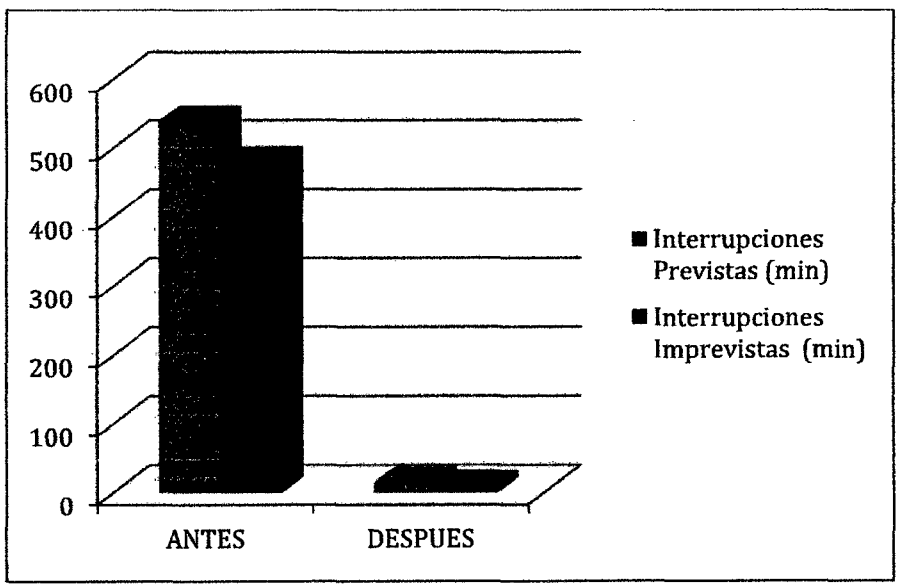



Gráfico 15. Interrupciones Previstas e Imprevistas (Antes, Después).

Luego se aplica un análisis de estadística descriptiva de la columna **DIFERENCIA** para hallar la ***Media*** y la ***Desviación Estándar.***

| *DIFERENCIA* | |
|---|---|
| | |
| Media | 497.5 |
| Error típico | 27.5 |
| Mediana | 497.5 |
| Moda | |
| Desviación estándar | 38.890873 |
| Varianza de la muestra | 1512.5 |
| Curtosis | |
| Coeficiente de asimetría | |
| Rango | 55 |
| Mínimo | 470 |
| Máximo | 525 |
| Suma | 995 |
| Cuenta | 2 |
| Nivel de confianza(95.0%) | 349.42063 |

Luego, para obtener el estadístico de prueba, se aplicará la siguiente fórmula.

$$t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}} = 18.090909$$

Cálculo estadístico de la prueba t para dos muestra emparejadas:

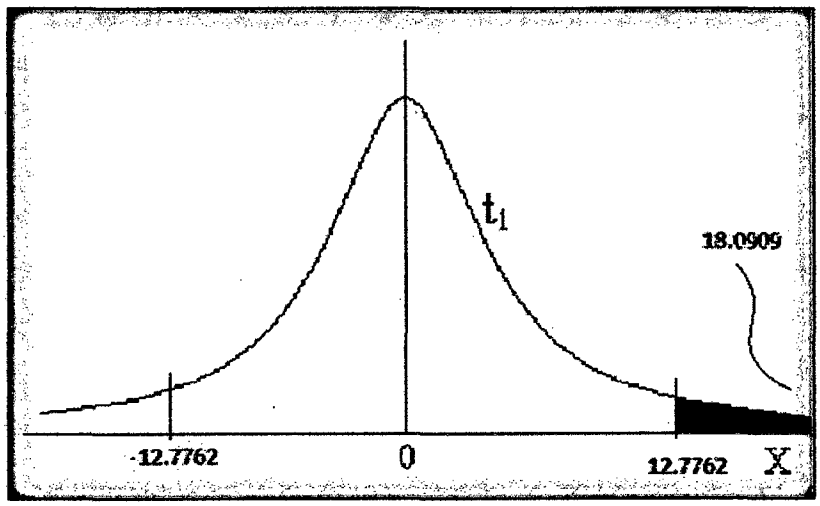
| Prueba t para medias de dos muestras emparejadas | | |
|---|---|---|
| | | |
| | *ANTES* | *DESPUES* |
| Media | 510 | 12.5 |
| Varianza | 1800 | 12.5 |
| Observaciones | 2 | 2 |
| Coeficiente de correlación de Pearson | 1 | |
| Diferencia hipotética de las medias | 0 | |
| Grados de libertad | 1 | |
| Estadístico t | 18.0909091 | |
| P(T<=t) una cola | 0.01757713 | |
| Valor crítico de t (una cola) | 6.31375151 | |
| P(T<=t) dos colas | 0.03515426 | |
| Valor crítico de t (dos colas) | 12.7062047 | |

***d. Cálculo del valor crítico de la estadística de prueba***

De acuerdo a los datos obtenidos, aplicando la prueba *t* para dos medidas de muestras emparejadas y usando $\alpha = 0.05$, tenemos como valor crítico:

$$-t_{0.03515426} \leq -12.7762 \text{ y } t_{0.03515426} \geq 12.7762$$

***e. Definición de la regla de decisión***



| | |
|---|---|
| **Hipótesis** | $H_0$: μd <= 0<br>$H_a$: μd > 0 |
| **Estadístico de prueba** | $t = \frac{X_d - \mu_d}{\frac{s_d}{\sqrt{n}}}$ |
| **Regla de rechazo:**<br>**Método *p* - value** | Rechazar $H_0$ si *p* - value ≤ α |
| **Regla de rechazo:**<br>**Método del valor crítico** | Rechazar $H_0$ si $t \leq$ - 12.7762 o<br>Rechazar $H_0$ si $t \geq$ 12.7762 |

***f. Toma de decisión de aceptar o rechazar Ho***

Como el *p* - value = 0.03515426 es menor que el nivel de significancia α = 0.05, y como el valor del estadístico de prueba *t* = *18*.0909091 es mayor que el valor crítico = 12.7762. Se tiene evidencia suficiente para rechazar la hipótesis nula.

Por lo tanto, se acepta la hipótesis alterna $H_a$: **La implementación de un clúster de alta disponibilidad disminuye el tiempo de inactividad del servicio de acceso a información.**

**4.2. Análisis de resultados.**

- La necesidad de la implementación de un clúster de alta disponibilidad en la empresa MINSE SENSE SOLUTIONS resulta más que necesaria ya que el sistema implementara tiene una vital importancia en las operaciones del proyecto minero Hudbay, la arquitectura de servidores que respalden al sistema tiene que ser de alto rendimiento y con un nivel de escalabilidad que asegure que será adaptable a las condiciones y requerimientos que se presenten, en el pre test realizado se denota que el rendimiento del servicio de acceso a la información sin la implementación de un clúster de alta disponibilidad es reducido ya que no tiene un respaldo que garantice la continuidad, no se tiene eficiencia energética y económica es decir no se puede migrar los equipos en caliente a un nodo y apagar otro por razones energéticas o de mantenimiento, el tráfico de información es alto y las

transacciones requeridas al servidor son muy altas y hacen que el rendimiento del CPU sea muy alto lo cual reduce el tiempo de vida del servidor, no tiene una transferencia de información y todo tipo de servicio por la red de forma rápida e ininterrumpida, la velocidad de respuesta puede disminuir al incrementar el número de transacciones, el tiempo de inactividad sería muy alto al tener que prescindir del servicio de información del servidor cuando este tenga que entrar en mantenimiento o ante el suceso de alguna interrupción no programada. Todo lo antes mencionado comprueba que se ha cumplido con el primer objetivo de esta investigación: **Caracterizar el estado actual y las necesidades de la disponibilidad de la información en la Empresa MINE SENSE SOLUTIONS SAC.**

- El segundo objetivo de esta investigación fue: **Diseñar un clúster de servidores para obtener Alta Disponibilidad de la Información en la Empresa MINE SENSE SOLUTIONS SAC.** Para cumplir con este objetivo se diseñó una arquitectura de servidores que pueda ofrecer una alta disponibilidad del servicio de información, para lo cual se escogió las herramientas que mejor se adecuan según los requerimiento antes mencionados, la base de datos utilizada fue **postgresSQL** que es motor de base datos más estable y robusto con el código abierto, para la replicación de la información se utilizó el mecanismo de réplica **Stream Replication** del mismo motor de base datos, para el manejo del clúster se escogió el middleware **Pgpool2** que se adapta de manera excelente a las necesidades requeridas, para el manejo de la alta disponibilidad del servicio del middleware se utilizó el sistema de alta disponibilidad por excelencia en Linux **HeartBeat.**

- El tercer objetivo alcanzado en esta investigación fue: **Validar el diseño propuesto que asegure la Alta Disponibilidad requerida por la empresa MINE SENSE SOLUTIONS SAC.** Este objetivo fue alcanzado luego de diseñar e implementar el clúster de alta disponibilidad y al hacer el comparativos del post test con el pre test se puedo lograr apreciar la mejora en el servicio de información con respecto al antes de la implementación del clúster, el nivel de rendimiento es óptimo gracias a que la arquitectura diseñada es resistente a caídas es decir en cuanto cae un nodo, el otro se encarga de todos los servicios, la arquitectura diseñada tiene eficiencia energética y económica es decir que es posible migrar los equipos en caliente a un nodo y apagar otro por

razones energéticas o de mantenimiento, la arquitectura de servidores tiene la capacidad para hacer frente a volúmenes de trabajo cada vez mayores, prestando así un nivel de rendimiento óptimo, además de transferir información y todo tipo de servicio por la red de forma rápida e ininterrumpidamente y mantener estable la velocidad de respuesta al incrementar el número de transacciones, el tráfico de información es distribuido entre los nodos del clúster y las transacciones requeridas al servidor se distribuyen y hacen que el rendimiento del CPU sea reduzca con lo cual reduce el tiempo de vida de los servidores. Todas estas características mencionadas hacen que la arquitectura diseñada tenga un alto rendimiento y un alto nivel de escalabilidad, balanceo de carga y una alta disponibilidad de información.

# CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

## 5.1. Conclusiones

- ✓ La necesidad de alta disponibilidad en el caso de estudio es evidente al dar un servicio de información a un sistema de vital importancia en las operaciones de la empresa, por lo cual fue necesario analizar y ver en mercado las mejores opciones y alternativas para el diseño e implementación del clúster que brinde la alta disponibilidad requerida.

- ✓ El diseño e implementación del clúster se dio en un entorno Linux, el cual es muy complejo en comparación con un entorno Windows, para lo cual es necesario tener conocimiento de lenguajes de programación y códigos establecidos para la consola que fue la principal herramienta para la implementación del clúster ya que por medio de esta se accede a los archivos y variables de configuración en la implementación.

- ✓ Los resultados obtenidos en el análisis de datos nos dan a relucir que la implementación del clúster brinda una disponibilidad de la información además de un mejor performance en el rendimiento del servicio teniendo características fundamentales para el óptimo desempeño del clúster como la escalabilidad y el balanceo de carga.

- ✓ La escalabilidad y el balanceo de carga permiten a nivel de hardware tener un mayor tiempo de vida de los equipos ya que la carga del tráfico de datos y las transacciones enviadas al servidor se distribuyen entre los nodos que lo componen.

- ✓ La arquitectura del clúster ha permitido alcanzar un aproximado de 99.99% de disponibilidad al año incluido las interrupciones previstas por mantenimiento de cada uno de los nodos que conforman el clúster.

## 5.2. Recomendaciones

- ✓ Analizar e investigar cada una de las variables y archivos de configuración que intervienen en la implementación del clúster con el fin de tener el conocimiento para poder actuar de la mejor manera ante un entorno cambiante de operaciones.

- ✓ Realizar una investigación en software propietario para un clúster de servidores en Windows para poder tener alta disponibilidad en instituciones que en determinados periodos de tiempo tienen saturación por la excesiva cantidad de usuarios accede a sus servicios.

- ✓ Tener alta disponibilidad a nivel de servidores y a nivel de dispositivos de interconectividad como routers y switch, ya que si éstos se convierten en un punto de falla único, no sirve de nada tener alta disponibilidad en el resto de dispositivos.

- ✓ Analizar todas las posibilidades de fallos para poder tener un plan de contingencia que asegure la continuidad del servicio de información después de un desastre.

## REFERENCIAS BIBLIOGRAFICAS


[1] B. B. A. S. GUSTAVO, "CONFIGURACIÓN DE UN CLUSTER DE ALTA DISPONIBILIDAD Y BALANCEO DE CARGA EN LINUX PARA SATISFACER GRAN DEMANDA WEB Y SERVICIOS DE RESOLUCIÓN DE NOMBRES," ESCUELA DE INGENIERÍA, ESCUELA POLITÉCNICA NACIONAL, Quito - Ecuador, 2007.

[2] A. R. Sapiña, "Alta disponibilidad en servidores y optimización de recursos hardware a bajo coste," Informática de Sistemas y Computadores, Universidad Politécnica de Valencia, Valencia - España, 2012.

[3] E. F. A. L. P. LEMUS, "SOLUCIÓN DE ALTA DISPONIBILIDAD DE BASE DE DATOS POR HARDWARE O POR SOFTWARE," ESCUELA DE INGENIERÍA EN CIENCIAS Y SISTEMA, UNIVERSIDAD DE SAN CARLOS DE GUATEMALA, Guatemala, 2005.

[4] J. W. B. Bedoya, "Proposición de un método para balanceo de carga en un cluster heterogéneo simulado en NS2," Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV), Universidad Nacional de Colombia Sede Medellín, Guadalajara, México, 2009.

[5] A. M. MÚNERA, "MÉTODO PARA EL MANEJO DEL BALANCEO DE CARGA EN SISTEMAS DE CÓMPUTO DISTRIBUIDO DE ALTO DESEMPEÑO," Magíster en Ingeniería – Ingeniería de Sistemas, ESCUELA DE SISTEMAS, UNIVERSIDAD NACIONAL DE COLOMBIA, MEDELLÍN - COLOMBIA, 2009.

[6] I. F. I. O. Martínez, "IMPLEMENTACIÓN DE UN SERVIDOR WEB VIRTUAL BALANCEADOR DE CARGA BASADO EN LINUX," FACULTAD DE TELEMÁTICA, UNIVERSIDAD DE COLIMA, Colima, Colombia, 2004.

[7] R. A. V. Rodriguez, "INFRAESTRUCTURA DE COMERCIO ELECTRÓNICO EN ALTA DISPONIBILIDAD," TÍTULO DE INGENIERO EN CIENCIAS Y SISTEMAS, Escuela de Ingeniería en Ciencias y Sistemas, Universidad de San Carlos de Guatemala, Guatemala, 2004.

[8] Y. A. a. C. Tutu, "From total order to database replication," presented at the International Conference on Distributed Computing Systems, 2004.

[9] P. N. Ayuso, "Arquitecturas para la alta disponibilidad de cortafuegos con estados," Departamento de Lenguajes y Sistemas Informáticos, UNIVERSIDAD DE SEVILLA, Andalucía - España, 2010.

[10] D. H. I. E. LUNA, "IMPLEMENTACIÓN DE UN CLUSTER DE ALTA DISPONIBILIDAD DE BASE DE DATOS PARA LA GEDGAPA," FACULTAD DE INGENIERÍA, UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO, Mexico, 2009.

[11] (2011, 21 DE MAYO DE 2011). *Cluster (informática).* Available: http://wwwclusterinformatica.blogspot.com/2011/05/cluster-informatica.html

[12] F. Wikimedia, "Clúster (informática)," 12 may 2014 2014.

[13] I. Corporation. (2013). *Funcionamiento de los nodos en un clúster.* Available: https://publib.boulder.ibm.com/tividd/td/TSMCW/GC32-0784 02/es_ES/HTML/anrwqs52121.htm

[14] F. Wikimedia, "Sistema operativo," 22 agosto 2014 2014.

[15] M. ROUAUX, "DISEÑO Y PROTOTIPO DE MIDDLEWARE BASADO EN CORBA PARA EL BUS CAN ", FACULTAD DE INGENIERÍA UNIVERSIDAD DE BUENOS AIRES, BUENOS AIRES, 2005.

[16] R. OSCAR, OpenMosix, and MPI, *High Performance Linux Clusters*: O'Reilly Media, 2004.

[17] (2007). *The High Availability Linux Project* Available: http://www.linux-ha.org/

[18] (2004). *Open Mosix version 1.0* Available: http://www.linuxvirtualserver.org/Documents.html

[19] B. M. G. JAVIER, "IMPLEMENTACIÓN Y CONFIGURACIÓN DE SERVIDORES APLICANDO CLUSTER PARA EL DESARROLLO DE PRACTICAS EN EL LABORATORIO DE SISTEMAS OPERATIVOS," INGENIERO EN SISTEMAS INFORMÁTICOS, ESCUELA DE INGENIERÍA EN SISTEMAS INFORMÁTICOS, UNIVERSIDAD TÉCNICA DE MANABÍ, MANABÍ – ECUADOR, 2012.

[20] (2013). *Conceptos de alta disponibilidad y recuperación ante desastres.* Available: http://technet.microsoft.com/es-es/library/jj715263(v=office.15).aspx

[21] F. Wikimedia. (2014). *Alta disponibilidad.* Available: http://es.wikipedia.org/wiki/Alta_disponibilidad

[22] F. Wikimedia. (2013). *Clúster de alta disponibilidad.* Available: http://es.wikipedia.org/wiki/Cluster_de_alta_disponibilidad

[23] J. P. Paredes. (205). *Alta disponibilidad para Linux.* Available: http://www.ibiblio.org/pub/linux/docs/LuCaS/Presentaciones/200103hispalinux/paredes/pdf/LinuxHA.pdf

[24] C. J. P. MATAMALA, "COSTO DE OPORTUNIDAD EN LA UTILIZACIÓN DE LOS SISTEMAS DE DESPACHO EN MINERÍA A CIELO ABIERTO," DEPARTAMENTO DE INGENIERÍA DE MINAS, UNIVERSIDAD DE CHILE, SANTIAGO DE CHILE, 2010.

[25] "Neumáticos para camiones mineros," in *La Tercera*, ed. Santiago de Chile, 2011.

[26] R. Martinez. (2009-2013 ). *Hot Standby y Streaming replication.* Available: http://www.postgresql.org.es/node/483

## ANEXOS

**Anexo 01.**

## CUESTIONARIO 01

| OBJETIVO |
|---|
| Objetivo del Cuestionario: El presente cuestionario tiene como objetivo poder determinar el nivel de rendimiento del servicio de acceso a la información. |
| **INSTRUCCIONES** |
| Analice cada uno de los criterios planteados y marque con un aspa el nivel de relevancia apropiado. |

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| MUY MALO | MALO | REGULAR | MUY BUENO | EXCELENTE |

| ITEM | CRITERIOS | MUY MALO | MALO | REGULAR | MUY BUENO | EXCELENTE |
|---|---|---|---|---|---|---|
| A | La arquitectura de servidores actual tiene resistencia a caídas (en cuanto cae un nodo, el otro se encarga de todos los servicios). | 1 | 2 | 3 | 4 | 5 |
| B | La arquitectura de servidores actual tiene eficiencia energética y económica. (migrar los equipos en caliente a un nodo y apagar otro por razones energéticas o de mantenimiento). | 1 | 2 | 3 | 4 | 5 |
| C | En la arquitectura de servidores actual, ante la caída de alguno de los ordenadores del clúster el servicio se puede ver mermado, pero mientras haya ordenadores en funcionamiento, éstos seguirán dando servicio. | 1 | 2 | 3 | 4 | 5 |
| D | En la arquitectura de servidores actual tiene la capacidad para hacer frente a volúmenes de trabajo cada vez mayores, prestando así un nivel de rendimiento óptimo. | 1 | 2 | 3 | 4 | 5 |
| E | La arquitectura de servidores actual puede transferir información y todo tipo de servicio por la red de forma rápida e ininterrumpidamente. | 1 | 2 | 3 | 4 | 5 |
| F | La arquitectura de servidores actual puede mantener estable la velocidad de respuesta al incrementar el número de transacciones. | 1 | 2 | 3 | 4 | 5 |

**Anexo 02.**

# CUESTIONARIO 02

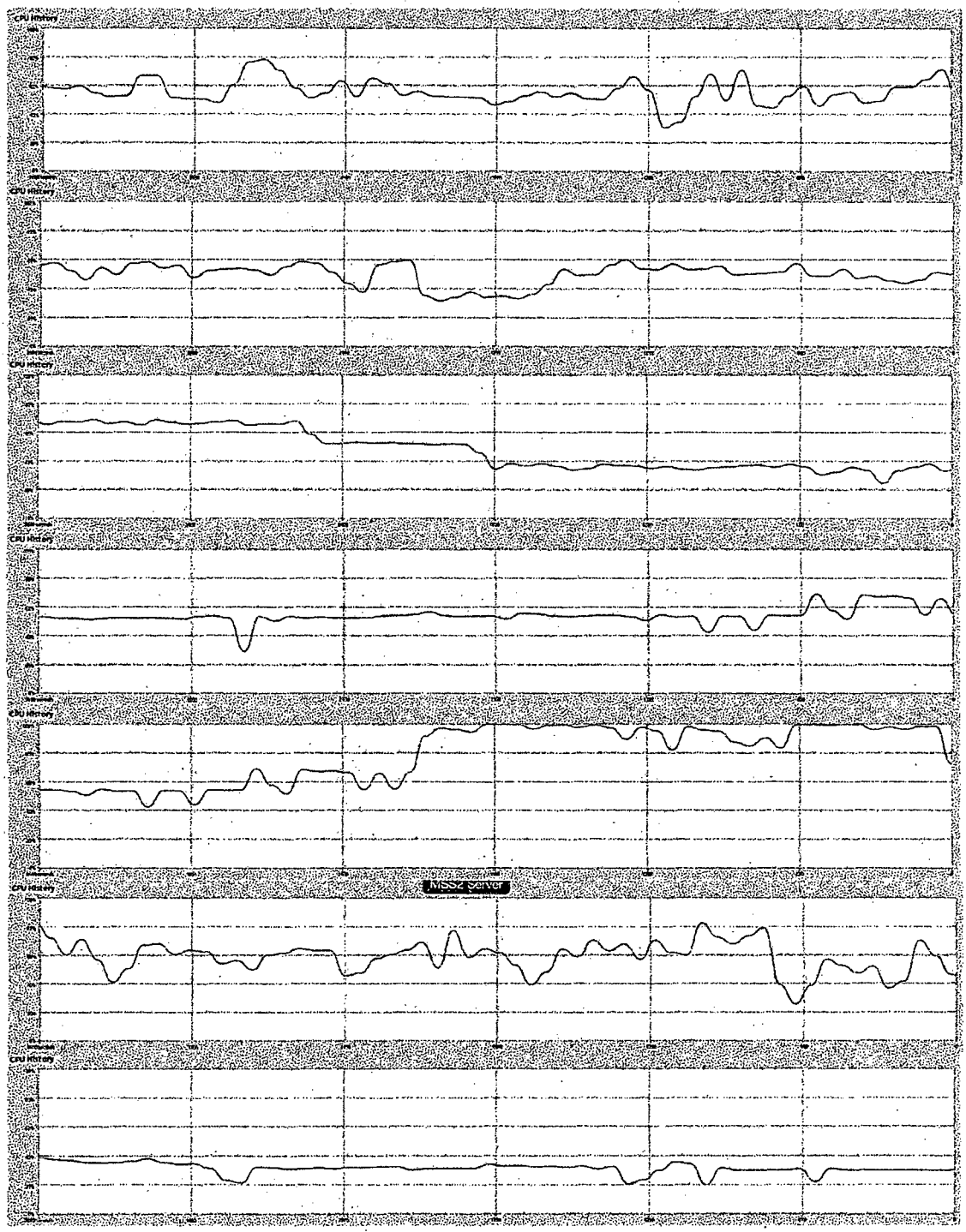
| OBJETIVO |
|---|
| Objetivo del Cuestionario: El presente cuestionario tiene como objetivo poder determinar el nivel de escalabilidad de la arquitectura actual del servicio de acceso a la información. |
| **INSTRUCCIONES** |
| Analice cada uno de los criterios planteados y marque con un aspa el nivel de relevancia apropiado. |

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| MUY MALO | MALO | REGULAR | MUY BUENO | EXCELENTE |

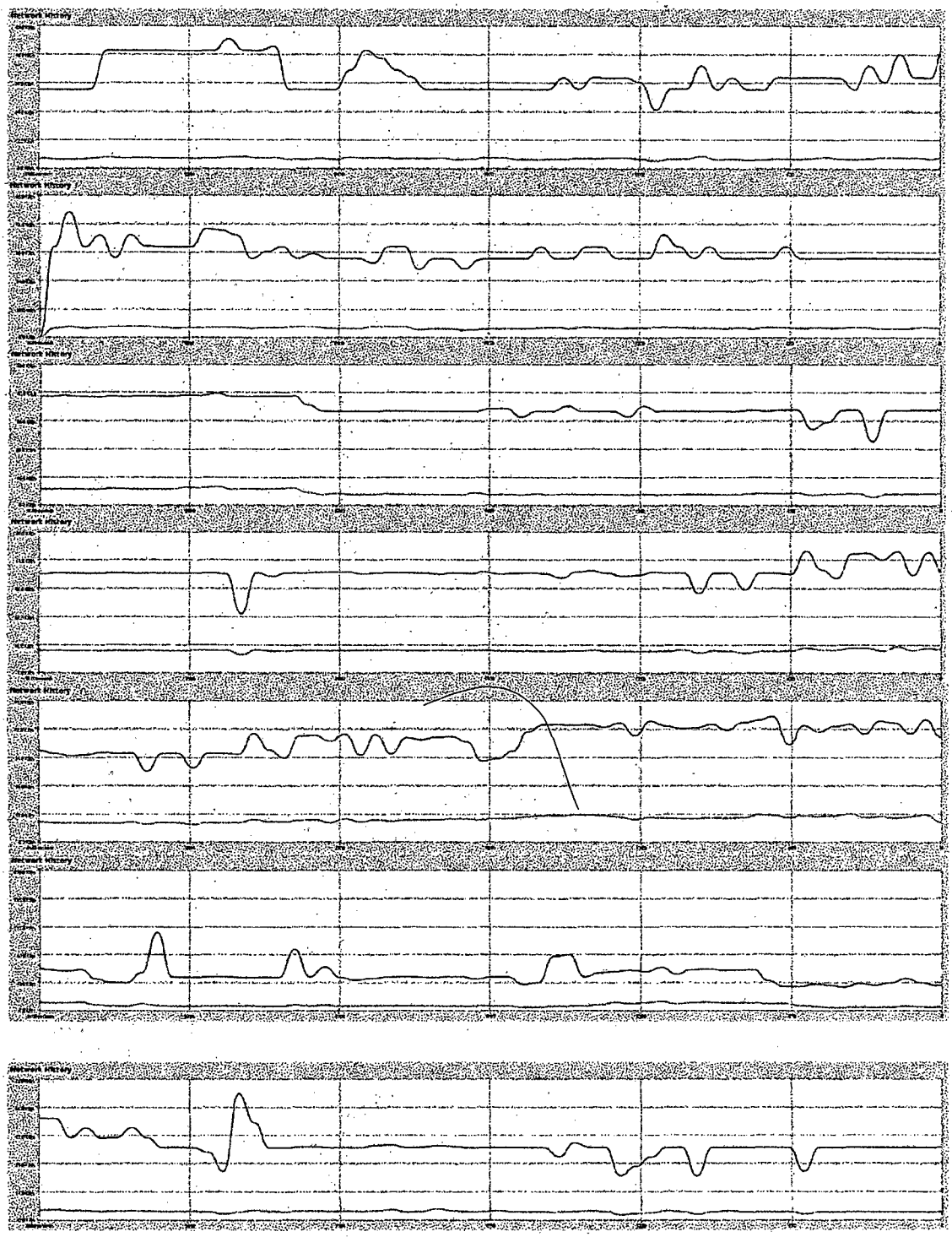
| ITEM | CRITERIOS | MUY MALO | MALO | REGULAR | MUY BUENO | EXCELENTE |
|---|---|---|---|---|---|---|
| A | La arquitectura de servidores actual está preparada para hacerse más grande sin perder calidad en los servicios ofrecidos. | 1 | 2 | 3 | 4 | 5 |
| B | La arquitectura de servidores actual tiene la capacidad de cambiar su tamaño o configuración para adaptarse a las circunstancias cambiantes. | 1 | 2 | 3 | 4 | 5 |
| C | La arquitectura de servidores actual puede acomodar a un número creciente de usuarios mediante la adición de hardware, sin tener que modificar el software. | 1 | 2 | 3 | 4 | 5 |
| D | La arquitectura de servidores actual tiene la capacidad de ampliar y reducir sus recursos para acomodar (a conveniencia), cargas más pesadas o más ligeras según se requiera. | 1 | 2 | 3 | 4 | 5 |
| E | La arquitectura de servidores actual puede ser utilizada para poder procesar más transacciones añadiendo por medio de nuevos procesadores, dispositivos y almacenamiento que se pueden implementar fácil y transparentemente sin apagados | 1 | 2 | 3 | 4 | 5 |
| F | La arquitectura de servidores actual tiene la habilidad de reaccionar y adaptarse al crecimiento continuo de trabajo de manera fluida. | 1 | 2 | 3 | 4 | 5 |
| G | La arquitectura de servidores actual puede ampliar su capacidad fácilmente añadiendo más ordenadores al clúster. | 1 | 2 | 3 | 4 | 5 |
| H | En la arquitectura de servidores actual tiene la capacidad de aumentar sus capacidades a través de mejores técnicas. | 1 | 2 | 3 | 4 | 5 |

## Anexo 03.

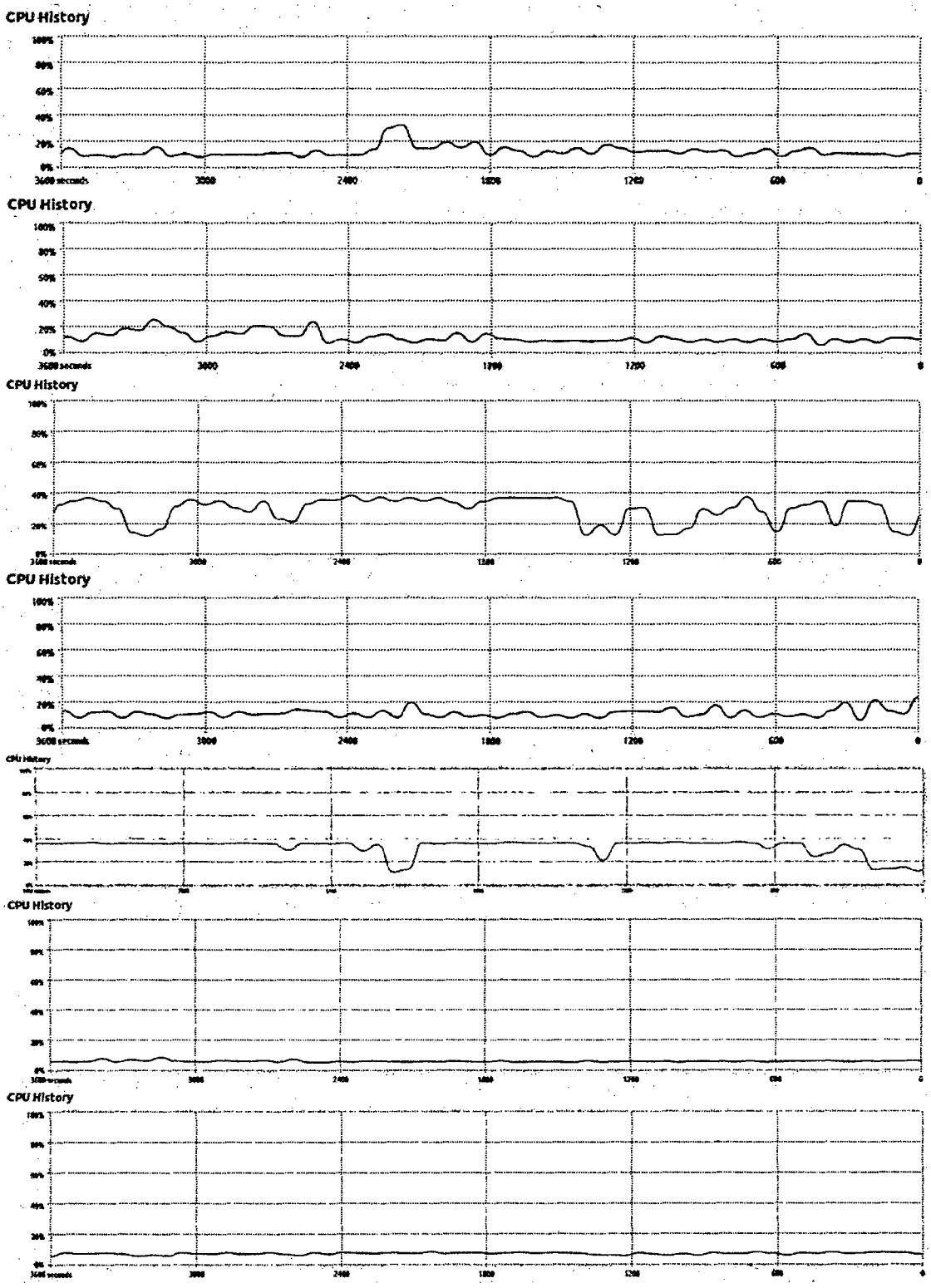
## Pre Test Rendimiento CPU (%).



**Pre Test Tráfico de Datos (KiB/s).**



## Post Test Rendimiento CPU (%).



## Post Test Tráfico de Datos (KiB/s).

**Anexo 04.**

## FICHA DE OBSERVACION.

| OBJETIVO |
|---|
| Objetivo de la Ficha de Observacion: La presente ficha de observacion tiene como objetivo poder determinar el tiempo de inoperatividad del servicio según las caidas de los nodos. |
| **INSTRUCCIONES** |
| Obtenga los tiempos de inclusion al cluster de cada uno de los nodos. |

| Nodo | Hora_inicio | Hora_fin | duración(seg) | tiempo estimado de inoperatividad(min) |
|---|---|---|---|---|
| HB1 | / / | / / | | |
| HB2 | / / | / / | | |
| HB3 | / / | / / | | |

**Anexo 05.**

## CUESTIONARIO

| OBJETIVO |
| --- |
| Objetivo del Cuestionario: El presente cuestionario tiene como objetivo estimar el tiempo de inoperatividad del servicio por causas previstas y no previstas. |
| **INSTRUCCIONES** |
| Analice cada uno de los criterios planteados y marque con un aspa. |

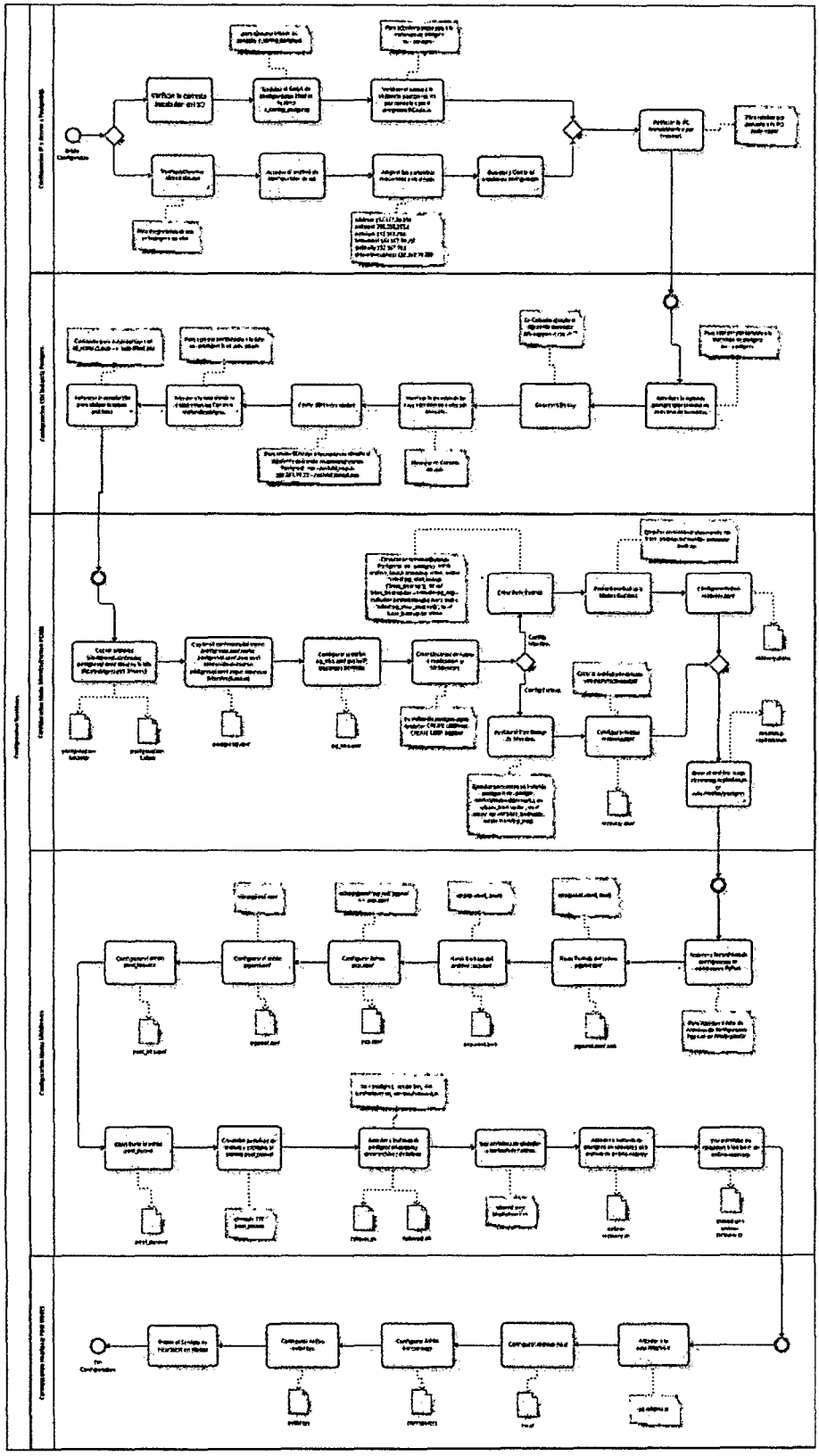
- ✓ Cantidad de interrupciones previstas al año.
  - o 2
  - o 3
  - o 4
  - o 5

- ✓ Cantidad de interrupciones imprevistas al año.
  - o 2
  - o 3
  - o 4
  - o 5

- ✓ Tiempo estimado de recuperación ante interrupciones previstas.
  - o 100 a 150 min.
  - o 150 a 200 min.
  - o 200 a 250 min.

- ✓ Tiempo estimado de recuperación ante interrupciones previstas.
  - o 100 a 150 min.
  - o 150 a 200 min.
  - o 200 a 250 min.

## Anexo 06. Proceso de Implementación del Clúster.



## Anexo 07. Solicitud de Validación Dirigida a Expertos.

**SOLICITUD DE VALIDACIÓN DIRIGIDA A EXPERTOS EN:**
**SISTEMAS DE OPTIMIZACION DE FLOTAS**

Cajamarca, 18 de Noviembre de 2014

Señor:
**Ing. Alejandro J. Ortiz Monteza.**
Presente

Reciba un cordial saludo

Motivado a su reconocida formación en materia de sistemas de optimización de flotas, me complace dirigirme a usted en solicitud de su valiosa colaboración para la validación de los cuestionarios y el registro de observación documental que anexo, los mismos servirán para recolectar información relativa a la investigación denominada: "Clúster de Servidores Linux para Alta Disponibilidad de la Información", que será presentado para optar el Título de Ingeniero de Sistemas otorgado por la Universidad Nacional de Cajamarca.

Asimismo, anexo el Instrumento para la validación de los cuestionarios, el registro de observación documental y el cuadro: Sistema de Variables e Indicadoras para una rápida comprensión y validación de los instrumentos.

Agradeciendo su valiosa colaboración en el desarrollo e impulso de la investigación, me suscribo.

Muy Cordialmente,

Lester Juan De Dios Villar Zamora.
DNI: 46874633
944442157

Recibido el 18 Noviembre 2014

Alejandro J. Ortiz Monteza
GERENTE DE OPERACIONES Y DESARROLLO
MINE SERVE SOLUTIONS S.A.C.

**Anexo 08. Instrumento para Validación de Encuestas.**

## INSTRUMENTO PARA LA VALIDACIÓN DE ENCUESTAS

| CRITERIOS | APRECIACIÓN CUALITATIVA | | | |
|---|---|---|---|---|
| | Excelente | Bueno | Regular | Deficiente |
| Presentación del instrumento | | X | | |
| Claridad en la redacción de los ítems | X | | | |
| Pertinencia de la variable con los indicadores | | X | | |
| Relevancia del contenido | | X | | |
| Factibilidad de la aplicación | X | | | |

Observaciones: ———

Validado por: Alejandro Ortiz Montero | DNI N°: 42515460

Profesión: Ingeniero de Minas

Lugar de Trabajo: Mine Sense Solutions

Cargo que desempeña: Gerente de Operaciones y Desarrollo

Lugar y fecha de validación: Pojpmoro 12 noviembre 2014

Firma:

Alejandro J. Ortiz Montero
GERENTE DE OPERACIONES Y DESARROLLO
MINE SENSE SOLUTIONS S.A.C.