

Portada

Universidad Nacional de Itapúa

Facultad de Ingeniería

HOJA DE EVALUACIÓN DE TFG

INTEGRANTES MESA EXAMINADORA:

-

-

-

-

-

CALIFICACIÓN FINAL:

Hugo Sendoa: _____ ()

David Krüger _____ ()

ACTA N°:

FECHA:

.....

Secretaria General

.....

Decano

Dedicatoria

Agradecimientos

Índice de Contenido

1. Marco Teorico

- a. Computacion de Altas Prestaciones en Paraguay
- b. Cluster
- c. Middleware SSI.
- d. Benchs para Base de Datos

2. Problematica

- a. Ejecución de aplicaciones de base de datos que requieran alta capacidad de procesamiento en la Facultad de Ingeniería.
- b. Alternativas a la adquisición de entornos que ofrezcan procesamiento de Altas Prestaciones.
- c. Uso de Cluster SSI como alternativa viable a un entorno de alta capacidad de procesamiento.

3. Solucion

- a. Análisis y diseño del cluster OpenSSI
- b. Benchs de performance para Base de Datos
 - OLTP
 - OLAP

4. Implementación del Cluster SSI

5. Benchs OLTP

- a. tpcc-uva

6. Benchs OLAP

- a. tpch

7. Resultados y Discusión.

- a. Planilla general de Resultados (todos los resultados Tabulados)
- b. Analisis de Resultados y graficos

8. Conclusiones y Recomendaciones

- a. Logros cumplidos
- b. Mejoras a lograr
- c. Futuras lineas de investigacion

9. Bibliografia

10. Anexos

- a. Instalacion del Sistema Operativo
- b. Instalacion del nodo master o init
- c. Agregar nodos a la infraestructura
- d. Instalacion de la herramienta visual de monitoreo.
- e. Bench TPCC-UVA
- f. Bench TPCh
- g. Script para instanciar el Postgres para pruebas OLTP
- h. Scripts utilitarios para pruebas OLAP
- i. Descripcion tecnica de Equipos
- j. Anexo de Formulario de Pruebas

Resumen

Introducción

Computación de Alto Rendimiento

La computación de Alto Rendimiento (HPC por sus siglas en ingles High Performance Computing) es un concepto que abarca un amplio espectro de la aún más extensa área de las Ciencias de la Computación, específicamente enfocado a brindar un sistema solido, claro, flexible y estructurado para garantizar el funcionamiento óptimo en la distribución de los recursos.

HPC esta relacionada con la “supercomputacion”, y surgio de los principios de este termino y trabajo, en ocaciones estos dos terminos son utilizados de manera indistinta. Sin embargo, el termino supercomputacion es usualmente aplicado a infraestructuras con propositos especificos, un subconjunto especifico de HPC. Las supercomputadoras suelen ser equipos mas poderosos y con sistemas especializados, y por lo tanto muy caros. Un ejemplo podria ser la supercomputadora Cray [[JE Beasley. Linear programming on Cray supercomputers. The Journal of the Operational Research Society. Vol. 41, No. 2, Feb., 1990](#)] o el sistema Thinking Machine [[Thinking Machines Connection Machine, Ultra High Speed Graphics Project-LANL](#)] de 1980. Sin embargo, estos sistemas de supercomputación especializados han sido casi completamente reemplazados por clusters de computadoras basicas no especializadas.

Las computadoras de CPU estandar estan basados en la arquitectura de Von Neumann [[John Backus. Can programming be liberated from the von Neumann style? Communications of the ACM, 21\(8\), 1977](#)], y basicamente cuentan con la restriccion de poder ejecutar unicamente una instrucción a la vez, una tras otra (ejemplo: procesamiento en serie). De esta manera, la unica opcion para poder realizar calculos mas veloces en este tipo de arquitectura era incrementando la velocidad con la que los calculos eran realizados. Aunque esta velocidad de calculo ha aumentado de forma constante y rapida desde la invencion del circuito integrado, el cual sigue la Ley de Moore [[Excerpts from A Conversation with Gordon Moore: Moore's Law. G Moore - Interview by Intel Corporation, Intel Corporation.](#)], muchos calculos deseables son todavia tan complejos y extensos que incluso en el equipo serial mas rapido, estos requiririan semanas o inclusive anos para finalizar el computo. Sin embargo, muchas

aplicaciones pueden dividirse en segmentos y estos segmentos resolverse simultáneamente en varios equipos. Cuanto mas se puede dividir la aplicación, mayor sera la ejecucion en paralelo. Todas las aplicaciones HPC involucran la paralización y ejecución del computo en múltiples entornos distribuidos.

Computacion Paralela y Distribuida

La computacion paralela permite que varios calculos sean ejecutados simultáneamente. Cuando los problemas complejos pueden ser segmentados en otros menos complejos que puedan ser resueltos concurrentemente (al mismo tiempo o en paralelo), en consecuencia es reducido el tiempo necesario para resolver dicho problema. Un punto importante en la paralización de aplicaciones es la frecuencia en que los segmentos de código necesitan comunicarse entre si. Algunos calculos pueden ser divididos en segmentos que unicamente necesitan una comunicación parcial del resultado entre ellos e inclusive pueden no ser necesaria la comunicación [Blaise Barney. *Introduction to Parallel Computing*. https://computing.llnl.gov/tutorials/parallel_comp/].

Cuando las subtareas necesitan comunicarse entre si para poder realizar el calculo en paralelo, un nuevo conjunto de cuestionamientos se introducen que complican la escritura de tales programas correctamente. Especificamente la necesidad de concurrencia y la sincronización introducen potenciales errores de programación, como podria ser la condicion de deadlocks (o poner bloqueos??) [E. G. Coffman, M. Elphick, A. Shoshani. *System Deadlocks*. *Journal ACM Computing Surveys (CSUR) Volume 3 Issue 2, June 1971*]. Estos problemas adicionales son el mayor obstaculo para la implementación de computacion paralela y obtener una buena prestacion paralela de un programa. Otra consideracion para la computacion paralela es que la sobrecarga de la comunicación puede y hara que se pierda todo el rendimiento en la ejecución simultanea de tareas. El incremento de velocidad de un programa es gobernado por la Ley de Amdahl [G. M. Amdahl. *Validity of the Single-Processor Approach to Achieving Large Scale Computing Capabilities*. In *AFIPS Conference Proceedings*, pages 483–485, April 1967], la cual que básicamente permite en teoría calcular la cantidad de aceleración que se puede obtener basada en la cantidad de cálculo que se puede en paralelo. En la actualidad, en donde los procesadores multinucleos ya son productos basicos, esta teoria requiere una mayor investigacion basandose en el rendimiento del

chip completo en lugar de centrarse en la eficiencia de cada núcleo [Mark D. Hill and Michael R. Marty. *Amdahl's Law in the Multicore Era. IEEE Computer* 2008].

Arquitectura de procesamiento en paralelo

Taxonomía para organizar las arquitecturas con el motivo de estudiarlas [M. Flynn. *Computer organizations and their effectiveness. IEEE Transactions on Computers, September 1972*], esta se basa en las instrucciones de control de secuencias de datos dispuestas en la arquitectura. Está dividida en:

- SISD (del inglés Single Instruction and Single Data): Tomando como ejemplo la Arquitectura de Von Neumann, corresponde a una única instrucción operando con una única secuencia de datos.
- SIMD (del inglés Single Instruction and Multiple Data): Una misma instrucción aplicada a múltiples secuencias de datos.
- MISD (del inglés Multiple Instruction and Single Data): Múltiples Instrucciones simultáneas aplicadas a una única secuencia de datos.
- MIMD (del inglés Multiple Instruction and Multiple Data): trata sobre las arquitecturas que permiten un múltiple acceso a secuencias de datos por instrucciones múltiples. Tratando de organizar esta última a finales de la década del 90 se procede a subdividirla en dos secciones [Ralph Duncan. *A Survey of Parallel Computer Architectures. IEEE Computer Society Press. Vol 23, Issue 2.*][K. Hwang, F. Briggs. *Computer Architecture and Parallel Processing. McGraw-Hill. 1994*]:
 - SMP (Symetric MultiProcessing): Arquitecturas que comparten una memoria física única y cuya velocidad disminuye según se incrementan los procesadores.
 - DMM (Distibuted Memory Multiprocessing): Son sistemas en las cuales el procesador posee su propia memoria, evitan problemas de bloqueos y evita

el acceso a la memoria física principal.

En el ámbito de HPC, en la mayoría de los casos se trata de sistemas MIMD, aunque los sistemas SIMD, en forma de clusters de computadores basados en GPU [[Zhe Fan, Feng Qiu, Arie Kaufman, and Suzanne Yoakum-Stover. GPU cluster for high performance computing. Proceedings of the 2004 ACM/IEEE Supercomputing Conference \(SC'04\), page 47059, November 2004](#)], se están convirtiendo en un subconjunto significativo y muy prometedor en la labor en la computación de alto rendimiento. Los sistemas MIMD pueden ser divididos en dos grupos principales, los que comparten una memoria principal (SMP), y aquellos que no comparten una memoria (DMM). Al momento de implementar una infraestructura de Cluster, usualmente se hace uso de ambos tipos de la arquitectura MIMD. Un procesador de memoria compartida, también llamado SMP (del inglés Symmetric multiprocessor), es aquel donde varios CPUs conviven en la misma máquina, un ejemplo de esto son los procesadores multinucleos.

HPC basado en el modelo de Cluster

Existen varias formas de presentar un modelo correspondiente al cómputo en altas prestaciones, como por ejemplo una estructura basada en el Modelo de Clúster. [agregar lo de hpcnotes waht is a cluster computer]. En forma general se define la palabra clúster como un número de elementos del mismo tipo agrupados entre sí. Su definición proviene del idioma inglés [19] y es utilizado para definir un conjunto de elementos, de igual manera Tomas Sterling lo señalaba en su libro [20] en el cual realiza analogías con los sistemas biológicos, organizaciones humanas y estructuras de computadores.

Un clúster brinda flexibilidad de configuración. El número de nodos, la memoria disponible por nodo, el número de procesadores por nodo y la topología de red utilizada para la interconexión de las partes son los parámetros más considerados a la hora de especificar el poder computacional del clúster. Otra característica muy bien vista por la comunidad tecnológica es la capacidad de adaptación a nuevas tecnologías de hardware y por tanto el incremento de la capacidad de cómputo no se vería ligado de forma unilateral a la compra de un nuevo equipo y de forma condescendiente el

desecho del anterior como sucede con los mainframes.

La clave del comportamiento del clúster se basa principalmente en el manejo de los planificadores de plazo [21] y en la capacidad de interconexión entre los componentes de la red conocidos también como nodos del clúster.

Características de un clúster.

Es un hecho que la mayoría de los más grandes problemas estratégicos en las cuales se aplica el cálculo numero en relación con las ciencias de la computación, solo puede ser resueltas mediante un maquina cuya ingeniería permita realizar un computo complejo.

De manera desafortunada para usuarios o empresas el costo de estos ordenadores no compensa la necesidad en un factor de costo o tiempo de uso. Es por ello que las alternativas viables para tener acceso a esta disponibilidad fueron siempre aceptadas y estudiadas.

El punto en el cual se unen las necesidades de disponibilidad, rendimiento y bajo costo de forma sinérgica dieron inicio a una tendencia que se expandió de acuerdo a la demanda de las necesidades. Es punto es conocido como Computación en Conjunto (del ingles Clúster Computing) cuya características principales se basan en la unión de una capa física de nodos, una abstracción del manejo del paso de mensajes y la aplicación de tecnologías Distribuidas.

Middleware SSI

El middleware puede ser considerado como una capa software que reside entre las aplicaciones y los niveles subyacentes como son el sistema operativo, las pilas de protocolo y el hardware [22]. Históricamente el concepto de middleware no proviene de la investigación académica, sino de la industria.

Una imagen de sistema único SSI (por sus siglas en ingles) es una propiedad de un sistema que oculta la distribución y heterogeneidad de sus recursos. y los vuelve disponibles a los usuarios y aplicaciones como un recurso computacional unificado. SSI provee a los usuarios una vista globalizada de los recursos disponibles en el sistema, desconsiderando al nodo al cual este asociado físicamente dentro del entorno distribuido. Además, SSI puede asegurar la continua operatividad del sistema luego de una falla (alta disponibilidad) así también garantiza que el sistema se cargue uniformemente, provea multiprocesamiento y gestión de los recursos.

Los objetivos de un diseño SSI para un sistema basado en Clúster se enfocan principalmente en la transparencia complete de la gestión de recursos, el rendimiento escalable, y la disponibilidad del sistema para el soporte de las aplicaciones para usuarios [19]. SSI puede ser definido como una ilusión creada por hardware o software, el cual presenta una colección de recursos como un único recurso mucho más poderoso [23].

Servicios y Beneficios de Middleware SSI

Los principales servicios de una imagen de sistema único SSI incluyen los siguientes [19]:

- Proporciona una visión simple y directa de todos los recursos y actividades del sistema desde cualquier nodo del clúster.
- Libera al usuario final de tener que saber en que parte del clúster se ejecutará la

aplicación.

- Permite el uso de los recursos de manera transparente con independencia de su ubicación física.
- Permite el trabajo del usuario por medio de una interfaz amigable y permite al administrador gestionar al clúster como una sola identidad.
- Ofrece la misma sintaxis de comandos como de otros sistemas y por lo tanto reduce el riesgo de errores de operaciones y con este resultado los usuarios finales observan un mayor rendimiento, fiabilidad y mayor disponibilidad del sistema.
- Permite centralizar/descentralizar la gestión del sistema y el control para evitar la necesidad de administradores expertos para la gestión del sistema.
- Simplifica de gran manera la gestión del sistema y reduce así el costo de propiedad.
- Proporciona la comunicación de mensajes sin dependencia de localización
- Beneficia a los desarrolladores de sistemas en la reducción del tiempo, esfuerzo y conocimientos necesarios para la realización de tareas que permiten al personal actual manejar grandes o más complejos sistemas.
- Promueve el desarrollo de herramientas estándares y servicios.

Un buen SSI se obtiene usualmente mediante la cooperación entre todos estos niveles, un nivel más bajo pueden simplificar la implementación de un ser superior.

SSI al nivel del sistema operativo

Los sistemas operativos de los Clúster soportan una eficiente ejecución de aplicaciones paralelas en entorno distribuido con aplicaciones secuenciales. El objetivo es reunir los recursos en un clúster para proporcionar un mejor rendimiento tanto para las aplicaciones secuenciales y paralelas.

Para lograr este objetivo, el sistema operativo debe de ser compatible con la programación planificada en masa de algoritmos paralelos [24], identificación de los recursos ociosos en el sistema, tales como procesadores, memoria y redes; además ofrecen acceso globalizado a ellos. Debe soportar de manera optima el proceso de migración para proporcionar equilibrio de carga dinámica, así como la comunicación entre procesos rápidos, tanto par alas aplicaciones del sistema y de nivel de usuario. El sistema operativo debe asegurarse de que estas características estén disponibles para el usuario sin la necesidad de llamadas adicionales al sistema.

A continuación se listan os sistemas operativos más representativos que soportan SSI al nivel de kernel:

SCO Unix Ware Non Stop Cluster

Es el software de alta disponibilidad de SCO. Amplia significativamente el soporte por hardware, por lo que es más fácil y menos costoso de implementar el software de clustering mas avanzado para sistemas Intel. Es una extensión para el sistema operativo UnixWare en el cual todas las aplicaciones se ejecutan de manera eficaz y fiable dentro de un entorno SSI el cual elimina toda la gestión de carga. Cuenta con IP estándar como la interconexión, eliminando la necesidad de algún hardware propietario. La arquitectura clúster de Unix Ware Non Stop ofrece soporte integrado para aplicaciones de conmutación por error mediante un enfoque de $n + 1$. Con este enfoque, la copia de seguridad de la aplicación puede ser reiniciada en cualquiera de los varios nodos dentro del clúster. Esto permite que un nodo actúe como un nodo de copia de seguridad para los demás nodos dentro del clúster [25].

Sun Solares-MC

Es una extensión prototipo de un kernel Solaris de nodo único. Proporciona imagen de sistema único y alta disponibilidad al nivel del kernel. Solaris MC esta implementado mediante técnicas de orientación a objetos. Utiliza de manera amplia el lenguaje de programación orientado a objetos de C++, el modelo estándar de objeto y el lenguaje de definición de interfaz de COBRA. Solaris MC utiliza un sistema de archivos global

llamado Proxy del Sistema de Archivos PXFS (las siglas en ingles de Proxy FileSystem). Las características principales incluyen la imagen de sistema único, la semántica coherente y de alto rendimiento. El PXFS hace que los accesos a ficheros se vuelva transparente para los procesos. PXFS logra una imagen de sistema único por medio de la interceptación de las operaciones de accesos a archivos a nivel de la interfaz vnode/VFS (WritingFileSystem, instancias para la operatividad de los FileSystems). Solaris MC asegura que las aplicaciones de red no necesiten ser modificados y observen la misma conectividad de red, independiente de en que nodo se este ejecutando la aplicación [26].

GLUnix

Otra alternativa disponible para que el sistema operativo soporte SSI es la implementación de una capa superior en el sistema operativo existente, la cual realiza las asignaciones globales de los recursos. Este es el enfoque seguido por GLUnix de Berkeley. Esta estrategia vuelve portable al sistema operativo y reduce el tiempo de desarrollo. GLUnix es una capa del sistema operativo diseñado para proveer soporte en la ejecución remota transparente, trabajos paralelos y secuenciales, balanceo de carga y compatibilidad con versiones anteriores de binarios de aplicaciones existentes. GLUnix esta completamente implementado en el nivel de usuario y no necesita modificación en el kernel, por lo que se vuelve mas fácil de implementar. Las principales características proporcionadas por GLUnix incluyen técnicas de planificación en paralelo de algoritmos paralelos; detección de recursos ociosos, proceso de migración, y balanceo de carga; comunicación rápida a nivel de usuario, y soporte de disponibilidad [27].

MOSIX

Mosix es una extensión del kernel de Linux que permite ejecutar aplicaciones no paralelizadas en un Clúster. Una de las posibilidades de MOSIX es la migración de procesos, que permite trasladar los procesos de nodo en nodo. Mosix opera de manera inadvertida y sus operaciones son transparentes para las aplicaciones. Esto significa que se pueden ejecutar aplicaciones secuenciales y paralelas al igual que lo haría un SMP (Sistema de Multiprocesamiento simétrico, Symmentricmultiprocessing). No es

necesario prestar atención en donde el proceso se está ejecutando o lo que otros usuarios podrían estar realizando. Poco después de que un nuevo proceso haya sido creado, Mosix intenta asignarlo al mejor nodo disponible en ese momento. Luego de esto, Mosix continúa monitoreando el nuevo proceso, así como también los otros procesos existentes, evaluando los nodos para maximizar el rendimiento global. Todo esto es realizado sin necesidad de alterar la interfaz de Linux. Esto significa que todos los procesos pueden ser monitoreados y controlados como si estuviesen ejecutándose en un nodo en particular. La última versión de Mosix, llamada MOSIX2 es compatible con Linux 2.6. MOSIX2 es implementada como una capa virtualizada del sistema operativo, lo que provee al usuario y a las aplicaciones una imagen de sistema único SSI [28].

OpenMosix

El sistema OpenMosix está basado en Mosix, la principal diferencia, se encuentra en su licencia GPL. OpenMosix es un conjunto de parches al kernel y unas utilidades y bibliotecas de área de usuario que permiten tener un sistema SSI completo para Linux. Al estar basado en el código de MOSIX, comparte algunas de sus características y limitaciones. OpenMosix hace uso del parche de Rik van Riel de mapeado inverso de memoria, que permite que el proceso que consume más recursos de OpenMosix pase de tener una complejidad computacional de $O(n)$ a una complejidad k . En la práctica, elimina una de las partes del código de OpenMosix que pueden consumir una cantidad apreciable de procesador. OpenMosix fue lanzado como un parche para el kernel de Linux, pero también estuvo disponible en Live CDs especializados. El desarrollo de OpenMosix ha sido detenido por sus desarrolladores, pero el proyecto LinuxPMI continúa su desarrollo por medio del código OpenMosix [29].

Kerrighed

Kerrighed es el resultado de un proyecto de investigación iniciado en 1999. Su objetivo es presentar un único SMP (Sistema de Multiprocesamiento simétrico, Symmentricmultiprocessing) por encima del clúster. Kerrighed se compone de un conjunto de servicios distribuidos del kernel a cargo de la gestión general de los recursos del cluster [29][30].

OpenSSI

OpenSSI surge a inicios del año 2001, esta basado en los proyectos Non-Stop Cluster de UnixWare [25]. Bruce Walker, director del proyecto y principal desarrollador de OpenSSI, demarco claramente que el objetivo era crear una plataforma para poder integrar tecnologías de clúster, con código abierto. La ultima versión de OpenSSI dispone de varios archivos del sistema y manejo de sistemas de disco en código abierto, como por ejemplo GFS, OpenGFS, Lustre, OCFS, DRBD, también integra un mecanismo de bloqueo distribuido (OpenDLM) y una política para obtener balanceo de carga, siendo esta una característica importante heredada de Mosix. OpenSSI permite el balanceo de carga de un clúster dinámicamente mediante el uso de migración de procesos, otra característica también heredada de Mosix. El modulo de migración de procesos de OpenSSI utiliza un mecanismo de acceso al recurso remoto manteniendo los recursos de acceso del sistema bloqueados para que estos no puedan ser migrados. Este mecanismo es usado principalmente en IPC (interfaz de la tarjeta de red), CFS (sistema de archivos del clúster) y también para algunas llamadas al sistema [29].

Benchmarks

Toda infraestructura o sistema debe ser evaluado para conocer las ventajas y/o inconvenientes que puedan presentarse. Para ello se pretende cuantificar los resultados obtenidos de modo a comparar el rendimiento. Existen varias formas de elaborar una evaluación de rendimiento y la tendencia es evaluar los elementos bajo situaciones ideales. En este capítulo se presentan los conceptos básicos y los resultados obtenidos, así como la metodología aplicada para la obtención de estos.

Se denomina benchmark al o los procesos que son ejecutados en una máquina para proveer una carga de trabajo significativa con el fin de medir el rendimiento. Existen diferentes formas de realizar un benchmark, de los cuales podemos destacar a las generadas por una aplicación, función o procedimiento específico [J. Dongarra. *Performance of various computers using standard linear equations software in a Fortran environment. ACM SIGARCH Computer Architecture News. Volume 11 Issue 5, December 1983*], la ejecución de un test iterativo de una subrutina [D. Bailey, J. Barton. *The NAS Benchmark Program. Numerical Aerodynamic Simulations Systems Division. 1986. <http://www.tjhsst.edu/~rlatimer/mpi/nas-doc.pdf>*] o programas sintéticos que analizan el rendimiento individual de cada componente en un ambiente real, brindando una noción del comportamiento [A. Bouteiller, T. Herault, G. Krawezik, P. Lemarinier, F. Cappello. *MPICH-V Project: A Multiprotocol Automatic Fault-Tolerant MPI. International Journal of High Performance Computing Applications* 2006; 20; 319].

Los benchmark miden el rendimiento y para ello deben satisfacer características consideradas como necesarias en el performance [D. Bailey, J. Barton. *The NAS Benchmark Program. Numerical Aerodynamic Simulations Systems Division. 1986. <http://www.tjhsst.edu/~rlatimer/mpi/nas-doc.pdf>*] como:

Capacidad de cálculo del procesador:

Independientemente, cada ordenador posee un procesador, cuyas características propias son incrementadas al formar parte de un conglomerado debido a

la distribución de carga de trabajo, no obstante ante excesivas cargas el microprocesador debe responder de forma óptima. Este debe ser contrastado con las especificaciones técnicas del hardware. En la actualidad existen muchas herramientas que nos ayudan a medir estas capacidades mediante operaciones matemáticas [L. Dongarra. *Introduction to the HPC Challenge Benchmark Suite*. 1995. <http://icl.cs.utk.edu/hpcc/pubs/>] [R. Numrich. *A note on scaling the Linpack benchmark*. Journal of Parallel and Distributed Computing. v.67 n.4. 2007]

Velocidad de I/O (del ingles Input/Output o entrada y salida):

En el común de las situaciones los sistemas que brindan rendimiento, trabajan con operaciones que requiere un espacio de memoria adicional debido a la complejidad de operaciones que dan como resultante la situación final, ejemplo de ello son las operaciones conocidas como Procesos de Transaccion en Linea o OLTP (por sus siglas del Ingles OnLine Transaction Processing) ampliamente utilizadas en el comercio electrónico y transacciones bancarias [S. Chen et all. *TPC-E vs. TPC-C: Characterizing the New TPC-E Benchmark via an I/O Comparison Study*. ACM SIGMOD V39, i3. 2010].

La comparación de estas características medibles con respecto a otros valores brinda la percepción de mejor o peor rendimiento respecto al punto comparado.

La aplicación de benchmark se encuentra ligada a obtener métricas de software, infraestructura o un conjunto de ellas caracterizadas por las unidades que son consideradas para obtener un valor cuantitativo. Es por ello que dependiendo del uso dado, pueden variar las métricas o puntos de referencias establecidos por un estándar.

Cuando las pruebas de test son realizadas en su totalidad, provee información que debe ser estudiada. El análisis de esta permite conocer las características y el comportamiento de los componentes ante situaciones similares. Este proceso es conocido como caracterización de pruebas. [J. Garcia. *Proyecto Fin de Carrera: Una técnica para la caracterización de nodos en Redes de Sensores Inalámbricas*. Escuela Técnica Superior de Ingeniería de Telecomunicación de la Universidad Politécnica de Cartagena. Julio de 2008].

La comparación de estas características medibles con respecto a otros valores brinda la percepción de mejor o peor rendimiento respecto al punto comparado.

La aplicación de benchmark se encuentra ligada a obtener métricas de software, infraestructura o un conjunto de ellas caracterizadas por las unidades que son consideradas para obtener un valor cuantitativo. Es por ello que dependiendo del uso dado, pueden variar las métricas o puntos de referencias establecidos por un estándar.

Cuando las pruebas de test son realizadas en su totalidad, provee información que debe ser estudiada. El análisis de esta permite conocer las características y el comportamiento de los componentes ante situaciones similares. Este proceso es conocido como caracterización de pruebas. [*J. Garcia. Proyecto Fin de Carrera: Una técnica para la caracterización de nodos en Redes de Sensores Inalámbricas. Escuela Técnica Superior de Ingeniería de Telecomunicación de la Universidad Politécnica de Cartagena. Julio de 2008*].

Tipos de Benchmarks

De manera a clasificar los benchmark se encuentran:

Micro benchmark:

Sentencia o conjunto de instrucción dedicada a la evaluación de un solo componente del sistema. .[Measuring Computer Performance: A Practitioner's Guide. D. Lilja. Cambridge University 2005. Pag. 133-134]. Generalmente, al ser tan pequeño, cabe en la memoria y presenta evaluaciones erróneas debido al cacheo.[Using micro benchmark to evaluate System Performance. B. Bershad, et all. Schooll of computer science Carnegie Mellon university.] [Microbenchmark performance comparison of high-speed cluster interconnects. L. Liu et all. IEEE Computer Society. 2004.]

Toy Benchmark:

Son programas completos que realizan operaciones de manera dispersa, si este corresponde a una sección del nucleo del programa en un bucle se lo llama Program

Kernels. Sus limitaciones de ejecución no son consideradas como test completos y deben ser complementados. [Measuring Computer Performance: A Practitioner's Guide. D. Lilja. Cambridge University 2005. Pag. 133-134]

Benchmark Sintético:

Es un conjunto de aplicaciones en un solo programa, utilizado para medir partes de un sistema. Trabajan de manera directa con la sección a ser tratada y, contrariamente a los benchmark de aplicación, no reflejan un trabajo real debido a que trabaja de forma aislada, demostrando la situación ideal del objeto o sección del sistema en observación. [Storage Performance—Metrics and Benchmarks. Peter M. Chen. David A. Patterson. Computer Science Division, Dept. of EECS. University of California, Berkeley. pmchen@cs.Berkeley.EDU, pattn@cs.Berkeley.EDU] [A synthetic benchmark. H J Curnow and B A Wichmann. Central Computer Agency, Riverwalk House, London SW1P 4RT. National Physical Laboratory, Teddington, Middlesex TW11 0LW. Computer Journal, Vol 19, No 1, pp43-49. 1976]

Benchmark de Aplicación:

provee un conjunto de programas o instrucciones estándares que se encargan de simular una carga de trabajo completa, de manera tal como el sistema se comportaría ante una situación cercana a la realidad. Este tipo de benchmark provee una medida de variables durante su ejecución y por ende es importante aislar estos elementos que son considerados como objeto de estudio. [Measuring Computer Performance: A Practitioner's Guide. D. Lilja. Cambridge University 2005. Pag. 133-134]

Aplicación de Benchmarking a Base de datos

La aplicación de Benchmark a Base de datos es un área en el cual se desarrollan varias investigaciones debido a su importancia en incrementar las prestaciones de las operaciones diarias de negocios (conocido como Business Operation), así como también en las proyecciones de mercado (o Business Intelligence). La gestión de las múltiples transacciones pequeñas bajo el esquema de OLTP, en contraste de las complejas consultas históricas utilizadas por el esquema OLAP (por sus siglas del

ingles On Line Analytical Processing o Procesamiento analítico en línea) permiten tener un amplio panorama en todas las operaciones comúnmente utilizadas en la administración de datos

Si la aplicación de un benchmark es dada a una base de datos se deben tener en cuenta los criterios de:

Capacidad de cálculo:

Específicamente las operaciones de punto flotante son las más resaltantes debido a su complejidad.

Interbloqueo de transacciones:

El tiempo de bloqueo entre operaciones que transaccionan en un mismo instante influye considerablemente en cargas de trabajos de magnitud.

Velocidad de Acceso a Disco (I/O):

Debido a que se manejan datos históricos y por la magnitud de estos no pueden ser alojados en memoria dinámica y por ello deben ser alojados en medios de almacenamientos. Para datos del esquema OLTP suelen manejarse de manera interna, en cambio para el esquema OLAP los datos pueden ser alojados externamente a la infraestructura principal [1. [Szepkuti. Difference Sequence Compression of Multidimensional Databases. http://arxiv.org/abs/1103.3857 . 2011](http://arxiv.org/abs/1103.3857)].

Se considera como objeto de validación la aplicación de un benchmark al banco de datos en diferentes situaciones de manera a verificar el comportamiento de la Base de Datos.

El resultado de estas pruebas define el comportamiento de la base de datos en diferentes escenarios contrastándolos con la infraestructura SSI.

Necesariamente la verificación de una base de datos en busca de rendimiento se debe

realizar emulando el trabajo regular en situaciones que se presentan de manera frecuente, es por ello que todo benchmark aplicado debe realizar sentencias SQL sobre la estructura en cuestión.

Varias organizaciones realizaron una definición para ser considerada como estándar y proveen formas de medir y reportar los resultados arrojados. Entre ellas podemos encontrar los estándares como SPEC [www.spec.org], TPC [www.tpc.org], Perfect Club [], The Open Source Database Benchmark [<http://osdb.sourceforge.net/>], The NAS Parallel Benchmark [<http://www.nas.nasa.gov/Resources/Software/npb.html>], HPL [<http://www.netlib.org/benchmark/hpl/>] y otros.

El Problema

Un banco de datos es conocido como una concentración de datos organizados de forma tal que la información se mantenga atómica, fiable y permita un acceso rápido a ella.

Los sistemas de altas prestaciones brindan optimización de recursos para ser aprovechados en procesos de manera a que sean procesados de forma eficiente, eficaz y confiable.

El Centro de Investigaciones de la Facultad de Ingeniería de la Universidad Nacional de Itapúa en sus diferentes líneas de investigación realiza estudios con el fin de ser ente generador de conocimiento en post de la investigación científica. Impulsado por docentes y alumnos llevan adelante proyectos en donde los bancos de datos son algunas de las herramientas más importantes del uso cotidiano.

Consideramos que la necesidad de contar con los elementos que faciliten y promuevan a la investigación científica, además de ser teóricos, deben de ser aplicados y para ello contar con la presencia de una infraestructura que lo permita es altamente necesario. Es por ello que la ejecución de aplicaciones de Banco de Datos en infraestructura de altas prestaciones es un factor ausente y necesario para la comunidad educativa.

Ejecución de aplicaciones de banco de datos que requieran capacidad de procesamiento en la Facultad de Ingeniería.

En la actualidad el Centro de investigaciones de la Facultad de Ingeniería tiene líneas de altas prestaciones y minería de datos que no cuentan con una

estructura que brinde un rendimiento optimizando el tiempo de proceso de las operaciones realizadas sobre banco de datos. Si bien muchas de las investigaciones se basan en tendencias tecnológicas, hasta la fecha estas bases teóricas no pueden ser aplicadas sobre entornos distribuidos complementados con prestaciones que garanticen la ejecución de los mismos.

Alternativas viables de Solución.

La elaboración de alternativas de solución no solo comprende en obtener todas las formas de resolución de la problemática, se basa en obtener un abanico de posibilidades que cumplan con el objetivo y además de ser viables se encuentren al alcance de los facilitadores.

El avance de tecnologías en microprocesadores y la reducción del tamaño del micro conductor permitieron que las empresas logren ordenadores específicos que provean prestaciones cercanas al óptimo buscado a altamente superior a la media de los demás ordenadores. Estos súper ordenadores son conocidos como Mainframes o Servidores dedicados. Definitivamente la incrementación del poder computacional vino acompañado de un incremento en el costo de los materiales y la tecnología utilizada para la elaboración de los mismos, es por ello que el costo de adquirir un centro de cómputo de estas características es considerado como una inversión que debe ser altamente justificada.

En la actualidad el auge de los sistemas distribuidos con tecnologías en las cuales computadoras separadas físicamente, pero unidas por un medio en el cual forman una red, permitieron dividir un problema de tamaño mayor a muchos problemas independientes y de menor tamaño, resolviendo de forma paralela partes de un todo.

Alternativas a la adquisición de entornos que ofrezcan procesamiento de Altas Prestaciones

La adquisición de estas alternativas que permitirían combatir la problemática relacionada a este trabajo se ven enfrentadas a las limitaciones presupuestales a la cual todo ente estatal se encuentra apegado. De esta manera, la adquisición de un mainframe como ordenador de computo para aplicaciones de banco de datos en el Centro de Investigación en Computación de la Facultad de Ingeniería, se vería obligado a competir con otras insuficiencias que son afrontadas por el ente y sujeto a una presupuesto con un periodo de aceptación y aprobación mayor al necesario para satisfacer las necesidades actuales.

La adquisición de una infraestructura por parte del Centro disminuiría las prestaciones con respecto a un mainframe, aumentaría considerablemente las prestaciones con respecto a un ordenador de capacidades medias y por sobre todo no estaría sujeto a un presupuesto elevado.

Otro ítem a ser considerado durante la elección de un entorno de procesamiento para el Centro de Investigaciones, se debe al nivel de criticidad de los datos que son tratados en las diferentes líneas de investigación con las cuales el CICFI se encuentra trabajando. Esta última razón inclina la balanza por la optar por una infraestructura que provea altas prestaciones al centro, debido a que los estudios que serán la razón de uso del entorno no poseen datos críticos.

Si bien las limitaciones físicas con las cuales cuentan el tipo de infraestructuras distribuida (como trafico de red y uso de espacio físico entre otros) no alcanzan las prestaciones de un servidor dedicado, permite lograr altas prestaciones para problemas complejos en los cuales un ordenador de características medias no tendría capacidad, se necesitan disminuir costos y mantener el nivel de criticidad de datos establecidos.

La Solución

Uso de Clúster SSI como alternativa viable a un entorno de Alta Prestaciones para Banco de Datos.

De manera a afrontar esta problemática se busco un punto de inflexión en donde se encuentra la brecha de la relación costo- rendimiento brindando servicios de manera transparente y cuyo resultado desemboca en las tecnologías distribuidas.

Por los antecedentes, la propuesta de proveer al Centro de Investigación en Computación de la Facultad de Ingeniería un entorno distribuido de bajo costo para procesos de Banco de Datos, se relaciona directamente con la unión de ordenadores físicos disponibles en el ente coordinados mediante una interconexión física y abstrayendo el paso de mensajes .

La utilización de entornos distribuidos de las características anteriormente mencionada requiere en la mayoría de los casos alterar la fuente de las operaciones, adaptándolos para el cálculo distribuido, es por ello que la propuesta de utilizar una capa que abstraee esa adaptación es incluida en esta propuesta.

En la unión de las necesidades y las disponibilidades nace esta propuesta basada en brindar al Centro de Investigación de la Facultad de Ingeniería un entorno distribuido con abstracción de paso de mensajes y un diseño que resulte transparente en la gestión de recursos, utilizando los ordenadores disponibles de la entidad.

Optando por OpenSSI como un clúster de alto desempeño y balanceo de carga

En los clústeres SSI de alto rendimiento los programas no tienen por que notar que se están ejecutando en uno de ellos. El único requisito es que los programas desplieguen múltiples procesos, los cuales se asignan de forma transparente entre los ordenadores del clúster. Aunque los clústeres SSI no son tan escalables como algunos otros tipos, ofrecen sin embargo la significativa ventaja de que los programas que se ejecutan en el sistema no tienen que conocer la existencia del clúster en el que se estén ejecutando. Otras plataformas de alto rendimiento basadas en clústeres requieren que el código fuente de los programas contengan código propio del clúster, o al menos que el programa se enlace con alguna librería de este. OpenSSI es una solución de clúster SSI extensa de código abierto para Linux basada en la plataforma clúster NonStop de HP. NonStop deriva de Locus, que se desarrollo en los ochenta.

OpenSSI puede repartir los procesos de forma transparente entre múltiples máquinas, característica conocida como nivelado de carga [31]. Otras plataformas de clúster SSI para Linux capaces de realizar el nivelado de carga son OpenMosix y Kerrighed. OpenMosix es el clúster SSI mas popular de Linux. En julio del 2007 se anuncio que el proyecto OpenMosix terminaría en marzo de 2008. Kerrighed es relativamente nuevo, encontrándose actualmente en una fase de desarrollo rápido.

OpenSSI monitoriza constantemente la carga en los ordenadores del clúster y migra automáticamente los procesos entre los nodos. Este sistema es capaz de migrar una aplicación multihilo, pero no es capaz de migrar hilos individualmente. Un proceso migrado puede continuar con las mismas operaciones que estaba realizando en la máquina original, puede leer y escribir en los mismos ficheros o dispositivos e incluso puede continuar una comunicación interproceso (IPC) sobre un socket.

La mayoría de las aplicaciones se ejecutan en OpenSSI sin ninguna modificación, con algunas excepciones, tal y como se indica en la pagina de ayuda del comando migrate. OpenSSI también proporciona una librería (API) que los programadores pueden utilizar para controlar el clúster.

Sabemos OpenMosix es el sistema de clustering más conocido y usado, sin embargo, aunque es difícil luchar contra un sistema plenamente acogido, desarrollado y mejorado por muchísima gente de diferente índole, OpenSSI, con cada actualización de Kernel nueva va ganando más adeptos y haciéndose un hueco entre el mundo de la supercomputación.

Características de OpenSSI

A continuación se muestran las características más importantes de OpenSSI:

- Manejo y administración del dominio de forma sencilla.
- Sistema de archivos de la raíz generado a través del clúster de forma sencilla.
- Copia sencilla de archivos binarios, librerías y administradores (como una contraseña).
- Se puede manejar los procesos por todo el clúster de manera sencilla según su PID.
- El nombre del clúster está disponible para todos los objetos del IPC.
- Contiene un dispositivo de acceso al clúster
- Nombrado de entidades consistente a lo largo de los nodos.
- Los archivos de acceso a todos los nodos son completos y públicos.
- La tecnología del sistema de archivos del clúster la tiene integrada y le provee flexibilidad y elección.
- Las interfaces del núcleo permitirán otra tecnología que esté desarrollada en código abierto.
- Está disponible un sistema de archivos del clúster con un failover público.
- El nombre y la dirección del clúster será altamente disponible, pudiéndose acceder a el siempre.
- La migración de procesos se puede realizar de forma completa incluyendo las llamadas al sistema.
- El balanceo de carga de un proceso se realiza en tiempo de ejecución.
- Incluye un conjunto de características de disponibilidad.
- Monitorización y reinicio del proceso.
- Servicio automático de failover.
- Sistema de archivos automáticos del failover.

-
- Dirección IP del clúster, manejo de la conexión y la habilidad de perder el nodo inicio sin matar a los procesos que están corriendo.
 - Garantía y un API para los miembros que usan una infraestructura de un Clúster bajo Linux.
 - Un API para migrar los procesos según los métodos: `rexec()` y `fork()`
 - Nodos que no necesitan discos gracias al arranque por red

Limitaciones de OpenSSI

En OpenSSI se debe de tener en cuenta ciertas limitaciones para que funcione correctamente, a continuación se muestra un listado de las limitaciones que se deben cumplir:

- Miembros del clúster: El número máximo de nodos en un clúster es de 125.
- El sistema de archivos de un clúster (CFS): Tamaño máximo de ficheros, número máximo de ficheros, direcciones y sistemas de archivo físicos inherentes.
- UID/GID: 16-bit en OpenSSI-1.2. 32-bit en OpenSSI-1.9.
- Tipos de dispositivos principales: 255 en OpenSSI-1.2. 4095 en OpenSSI-1.9.
- Tipos de dispositivos secundarios: Número máximo de puntos de montaje por tipo de sistema de archivos. 8-bit in OpenSSI-1.2. 20-bit in OpenSSI-1.9.
- IPC: Máximo número de semáforos compartidos.
- Sockets: Máximo número de sockets.
- Procesos: Máximo número de procesos. 32,000 en OpenSSI-1.2 y 1 billón(americano) en OpenSSI-1.9.
- PTY: Máximo número de pty's.
- HA-LVS: Máximo número de conexiones, de directorios, de CVIP. Igual que el original LVS, millones de conexiones.
- DRBB-SSI: Máximo tamaño del volumen por dispositivo de drbd, máximo de dispositivos drbd. Igual que el original DRBD.

Implementación del clúster OpenSSI

Como se habló en la sección **[Optando por OpenSSI como un clúster de alto desempeño y balanceo de carga]**, la distribución de procesos en los nodos del clúster es la principal característica de funcionalidad de OpenSSI. De una forma ideal, los procesos buscaran ejecutarse en un ambiente igual o muy similar del nodo donde provienen, tal vez con mejores recursos de memoria y procesador. Hablando en términos del programa, éste estaría ejecutándose en nodos externos, por lo que es indispensable que el procesador externo tenga el conjunto de instrucciones necesarias para poder procesar el código compilado en el nodo anfitrión, de otra forma el programa no podría ejecutarse.

Con base en estas observaciones, se concluye que la construcción de un clúster OpenSSI como en otros tipos de clústeres es muy recomendable que se haga con hardware homogéneo, aunque muchas de las veces debido a los recursos con los que se pueden disponer en nuestro centros de investigación no es posible cumplir con este requerimiento, no indispensable pero si recomendado, pues además de proveer un ambiente de ejecución adecuado para los procesos, facilita la administración del software instalado en los nodos del clúster.

Requerimientos para la implementación de OpenSSI

Requerimientos de Hardware

La elección del hardware a utilizar en un clúster puede definirse primeramente en base a conocer la magnitud del problema o problemas que se quieren resolver, el hardware que puede adquirirse en el mercado, los recursos económicos y humanos con los que se cuentan, tanto para administrar el clúster cómo para programar y ejecutar las aplicaciones. Cuando se planea la adquisición, se propone la compra del “mejor” equipo de computo.

La descripción de estos nodos es una propuesta ideal para ensamblar el clúster, en ésta

también se describe el tipo de red o canal de comunicación que se recomienda, aunque por supuesto es hardware que cumple con los requerimientos básicos. Cabe aclarar que se propone la construcción de un clúster específico, pues su propósito, problema o proyecto en particular, que se ha propuesto es para la ejecución de banco de datos.

En el capítulo **N**, en las pruebas de rendimiento se mostrará la utilidad de OpenSSI en relación a motores de banco de datos, por tanto se propondrá el diseño de un clúster que pueda en el mayor de los casos responder a las necesidades generales.

Se identifican como nodos aquellas unidades individuales de procesamiento en el clúster. Para la implementación de un clúster OpenSSI, el hardware soportado es para las arquitecturas IA32 y compatibles, es decir, en términos del hardware disponible en el mercado se refiere a los procesadores Intel y AMD con tecnología para procesamiento de palabras de 32 bits. OpenSSI ha sido desarrollado y probado para funcionar en estos dos tipos de procesadores y es recomendable no ensamblar clústeres con ambos tipos de procesadores.

Los factores en la elección de uno u otro tipo de procesador son principalmente su costo y también por supuesto el software que se podrá aprovechar al máximo. Como ejemplo, se puede mencionar que por cuestiones de mercado, los procesadores AMD han sido más económicos sin querer decir con esto que tengan un menor rendimiento o tecnologías más obsoleta o atrasada, pero en cuanto al desarrollo de aplicaciones de software comerciales y no comerciales, como por ejemplos los compiladores, estas han sido desarrolladas y optimizadas para los procesadores Intel. Esto no quiere decir que sea imposible de implementar un clúster de bajo costo con hardware de bajo costo, simplemente se requiere en la mayoría de los casos un estudio de las herramientas disponibles para optimizar las aplicaciones de los usuarios.

Para este trabajo de tesis, el hardware requerido se encuentra disponible en el Centro de Investigación de Computo de la Facultad de Ingeniería, C.I.C.F.I.; allí se implementará la infraestructura de un clúster conformado por 3 nodos. A continuación se listara el hardware disponible de manera ilustrativa, no se abarcara en detalles los dispositivos periféricos (teclado, mouse, etc.), estas quedan a criterio de las necesidades reales que se pueda dar a estos dispositivos en el clúster

Procesador	AMD Athlon(tm) Processor LE-1640 Processor Speed: 2.6 GHz. L2: 1 MB.
Memoria (RAM):	2 GB DDR-2 800 MHz.
Disco Duro (Hard Disk):	250 GB, SCSI de 15,000 rpm.
Placa Madre (Motherboard):	ASUSTeK Computer INC. M2N-MX SE Plus System Bus Speed: 800 MHz. System Memory Speed: 667 MHz.
Tarjeta de Red	Intel (R) Pro/100 Network Conecction Fast Ethernet

Requerimientos de Software

Sistema Operativos:

Debian GNU/Linux es un sistema operativo libre que soporta un total de doce arquitecturas de procesador e incluye los entornos de escritorio KDE, GNOME, Xfce y LXDE. La versión 5.0.8 Lenny con kernel 2.6 es compatible con las versiones openssi 1.9.6 y la versión alpha de openssi 2.0.

Sistema de Archivos:

Ext3: Sistema de archivo utilizado para las particiones Linux, incluye la característica de *journaling* que previene el riesgo de corrupciones del sistema de archivos y es de los mas utilizados por presentar mejor desempeño en el manejo de archivos.

Cluster File System (CFS): Sistema de archivos distribuidos pertenecientes a un grupo de servidores que trabajan en conjunto para proporcionar un servicio de alto

rendimiento para sus usuarios en vez de un único servidor con un conjunto de clientes. Para los usuarios del cluster el sistema de archivos es transparente, simplemente un sistema de archivos. El programa de manejo del sistema de archivos se encarga de distribuir las solicitudes a través de los elementos del almacenamiento del cluster. Los sistemas de archivos en cluster (CFS) permiten que múltiples servidores puedan acceder al mismo sistema de archivos. CFS resuelve las desventajas y complejidades de los discos duros proveyendo una solución más simple para la administración del almacenamiento

Herramienta de Monitoreo:

Openssi webView: Aplicación grafica de monitoreo de OpenSSI, con ella puede realizarse parte de las tareas comunes de monitoreo que se harían con los comandos. El propósito de esta herramienta es la de proveer una visión general del estado del cluster, graficar las funciones claves del sistema. Permite al administrador del cluster controlar el estado de los recursos de cada nodo y de la infraestructura en general.

Organización de los nodos

En esta sección se da una descripción de la topología de la red y de los componentes de la conexión entre nodos del clúster, como son el switch y el cable de red utilizados. Cabe mencionar que esta configuración es parte de la red actual del Centro de Investigación en Computación de la Facultad de Ingeniería (CICFI).

La topología de red utilizada es la estrella, se caracteriza por tener todos sus nodos conectados a un controlador central, en este caso, un switch de 8 nodos de los cuales 4 nodos pertenecen al clúster. Todas las comunicaciones pasan a través del switch, siendo éste el encargado de controlarlas. Por este motivo, el fallo de un nodo en particular es fácil de detectar y no daña el resto de la red, pero un fallo en el controlador central desactiva la red completa.

El cable con el que están conectados los nodos del cluster al switch que los une, es estructurado de la marca Kron, para velocidades de transmisión de

datos Ethernet 10/100/1000.

Los equipos conectados a este switch se conectan a una velocidad auto negociable de 10Base-T/100Base-TX, según la tarjeta de red.

Instalación del clúster OpenSSI

El proceso de instalación es un poco dificultoso al inicio, principalmente por lo incompleto y esparzo que resulta ser la documentación oficial[<http://openssi.org/cgi-bin/view?page=docs>]. Es bastante aconsejable estar muy bien familiarizado con la distribución que se estará utilizando, en este caso en particular Debian Lenny, de modo que se puedan investigar y solventar los posibles problemas.

En un entorno OpenSSI debe existir un nodo especial denominado nodo init o master. Éste arranca directamente desde el sistema de archivos raíz. El resto de los nodos arrancan desde la red. Para arrancar los nodos non-init o esclavos por red es necesario Etherboot (definiciones) o PXE (definiciones). Etherboot es software libre y se instala en un medio que permita el arranque, como un CD o un disco duro, o bien se graba en la memoria ROM de la tarjeta de red. La mayoría de las tarjetas de red son compatibles con Etherboot. PXE, por otro lado, se encuentra integrada en algunas tarjetas de red y normalmente se habilita desde el menú de la BIOS.

Debido a que el propósito principal de este trabajo de tesis, es proveer una guía metodología completa de la instalación del nodo master, agregar nodos esclavos y la instalación de una herramienta web para la monitorización visual del cluster OpenSSI.

A continuación se describirá los procesos para implementar un cluster OpenSSI en una distribución de Debian Lenny con los recursos de hardware disponibles en el C.I.C.F.I.

Instalación del Sistema Operativo Debian Lenny

Estas instrucciones asumen que se realizara una instalación limpia del sistema operativo, eso quiere decir que no se estará actualizando o modificando las configuraciones existentes de alguna de las computadoras. Para mas detalles de este

proceso, se provee el **Anexo 1. Instalando Debian** en donde se encuentran los pasos para este proceso.

1. Instalar Debian Lenny (<http://www.debian.org/releases/lenny/debian-installer/>) en el nodo master. No existe la necesidad de instalar esta distribución en los demás nodos.
2. Es recomendado que se realice la partición del disco duro usando las herramientas que provee durante la instalación. El sistema de archivos recomendado es ext3 debido a su capacidad transaccional. La tabla de particiones debería estar conformada de la siguiente manera: **/boot**, **/swap**, y **/**.
3. Configurar GRUB (<http://www.gnu.org/software/grub/>) como gestor de arranque. OpenSSI no soporta LILO (<http://lilo.alioth.debian.org/>).
4. Configurar las tarjetas de red con una dirección IP estática. Tanto la tarjeta de red para la conexión a una red externa, como la tarjeta de red para la interconexión con los demás nodos. Durante el proceso de instalación del OpenSSI se configurara la tarjeta de red para la interconexión con los demás nodos esclavos.

Instalación del OpenSSI

Los siguientes procedimientos son los necesarios para la instalación del kernel de OpenSSI en el nodo master. Estos procedimientos pueden encontrarse detallados en el **Anexo 2. Instalación el nodo init o master**.

1. Agregar las siguientes líneas al archivo `/etc/apt/sources.list`

```
deb http://deb.openssi.org/openssi 1.9.6-lenny-preview openssi extras
deb-src http://deb.openssi.org/openssi 1.9.6-lenny-preview openssi extras
```

2. Agregar las siguientes líneas al archivo `/etc/apt/preferences`

```
Package: *
Pin: origin deb.openssi.org
Pin-Priority: 1001
```

3. Si es necesario, configurar el http proxy.
4. Configurar la dirección IP estática para la tarjeta de red que servirá de interconexión para los nodos esclavos

```
#Setting for the Cluster Interconnect  
allow-hotplug eth1  
iface eth1 inet static  
address 192.168.64.1  
netmask 255.255.255.0  
broadcast 192.168.64.255
```

5. Ejecutar

```
# apt-get update  
# apt-get dist-upgrade
```

6. Agregar los instaladores necesarios de las tarjetas de red al archive /etc/mkinitrd/modules, los cuales serán usados durante el arranque de los nodos del cluster.

7. Ejecutar

```
#apt-get install openssi
```

Este comando instalará openssi y creará el primer nodo, init o master, del cluster. Durante la creación del primer nodo, mediante el comando *ssi-create*, aparecerán algunas preguntas relacionadas a las configuraciones del cluster. Para los detalles del mismo, se podrán obtener en la referencia al **Anexo 2. Instalación del nodo init o master**.

8. Reiniciar el nodo master

Agregar nuevos nodos

Una vez instalado el sistema operativo en el nodo master, instalado el kernel openSSI en el mismo, se procede a agregar nodos para poder contar con una infraestructura

Cluster. Los nodos esclavos son iniciados por medio de un método de inicio por red. Esto evita la necesidad de instalar el sistema operativo y el kernel del openssi en mas de un nodo. Para iniciar por red un nuevo nodo, es necesario que la tarjeta de red de los nodos esclavos soporten PXE o Etherboot. Estos procedimientos pueden encontrarse detallados en el **Anexo 3. Agregar nodos a la infraestructura**.

1. Si la tarjeta de red del nuevo nodo no posee soporte de inicio PXE, es necesario descargar el instalador Etherboot desde la siguiente url <http://rom-o-matic.net/gpxe/gpxe-1.0.1/contrib/rom-o-matic/> y generar un CD de inicio.
2. Si el nodo requiere un instalador que no se encuentra dentro del archivo `/etc/mkinitrd/modules` (en el nodo master), es necesario agregar el nombre del instalador en ese archivo y luego volver a generar el disco de inicio para que el cluster incluya ese instalador a su lista. Para realizar esto:

```
#mkinitrd -o <init RD image file> <kernel-version>  
# ssi-ksync
```

3. Conectar el nodo esclavo al switch conectado al cluster, insertar el CD de inicio previamente grabado e inicio el nodo. El CD mostrara la dirección MAC del nodo esclavo y espera un tiempo buscando un servidor DHCP que pueda responderle.
4. En el nodo master ejecutar el comando `ssi-addnode -hwaddress="MAC_ADDRESS"` en donde MAC_ADDRESS es la dirección MAC del nodo esclavo. Esto hará que el proceso de agregado de nodos inicie y solicitara las configuraciones, estas se pueden encontrar en el **Anexo 3. Agregar nodos a la infraestructura**.
5. El programa hará todo lo necesario para agregar el nodo al cluster. Es necesario esperar que el nuevo nodo se agregue. Un mensaje de "nodeup" aparecerá en la consola del primer nodo indicando el éxito de la operación. Puede comprobar el estado del cluster por medio del comando `cluster -v`.
6. Si se desea configurar el nuevo nodo como un posible nodo master, y habilitar la tolerancia a fallo (failover), es necesario realizar ciertas configuraciones en el

hardware del nodo tales como *ssi-chnode* y configurar las particiones del nuevo nodo así como el sistema de inicio GRUB.

7. Repetir los pasos anteriores cada vez que se desee agregar otros nodos al cluster.

Monitorización visual del clúster OpenSSI

Una vez lograda la instalación del nodo master, y agregado nodos esclavos, el siguiente paso es la instalación de la herramienta de monitoreo OpenSSI webView. OpenSSI webView es una herramienta PHP desarrollada por Kilian Cavalotti y extendida por el propio equipo de la tesis. Se encuentra disponible en esta dirección <http://darthvinsus.github.com/webview/>

La herramienta permite visualizar la carga en cada nodo del cluster, monitorizar el estado de cada nodo y ver las graficas con diversas estadísticas. También es posible monitorizar los procesos en cada nodo y migrar manualmente un proceso a otro nodo. Para utilizar esta herramienta es necesario realizar los siguientes pasos. Para mas referencias se puede consultar el **Anexo 4. Monitorización de OpenSSI**.

1. Extraer directamente en la carpeta del servidor el archivo descargado desde la dirección <http://darthvinsus.github.com/webview/>

```
# cd <DESTDIR>  
# tar xvfj DarthVinsus-webview-1902726.tar.gz
```

2. Establecer los permisos adecuados en el directorio de gráficos, para la generación de gráficos y la recolección de datos en el archivo config.php, para poder modificar y guardar la configuración desde la interfaz web.

```
# cd <DESTDIR>/openssi-webview  
# chown -R <USER> graphs  
# chown <USER> config.php
```

3. Agregar al cron la tarea de mantener actualizada la interfaz a través de la recolección de datos por medio de un script.

```
*/5 * * * * <USER> [ -d <DESTDIR> ] && (cd <DESTDIR> && ./graphs/update_all.sh  
)
```

4. Habilitar la migración de nodos por medio de la interfaz web, para ello modificar el archivo *etc/sudoers* y permitir la migración de procesos a todos los usuarios.

```
<USER> ALL = NOPASSWD: /usr/bin/migrate
```

Una vez realizado los pasos, acceda desde algún navegador web de su preferencia a la dirección *http://IP_NODO_MASTER/webview*, donde *IP_NODO_MASTER* es la dirección IP del nodo master, configurado en la sección

Benchmarks OLTP

TPCC-uva

Benchmarks OLAP

TPCh

Resultados y Discusión

Planilla general de los resultados

Análisis de resultados y gráficos

Conclusiones y Recomendaciones

Logros cumplidos

Mejoras a lograr

Futuras líneas de investigación

Bibliografía

Anexos

Instalación del Sistema Operativo Debian

La primera parte de la presente guía metodología para implementar una infraestructura cluster SSI consiste en instalar el nodo raíz. Para ello se utilizara la versión 5.0.8 de Debian, conocido como Lenny. Recomendada la version netinstall del mismo que se puede obtener desde la siguiente dirección.

```
1 http://www.debian.org/releases/lenny/debian-installer/
```

Netinstall significa que únicamente el sistema base se encuentra en la imagen .iso, los demás servicios se podrán descargar de internet posteriormente. Se recomienda que la partición este vacía, si es la primera vez que va a realizar una instalación Linux. Grabe el ISO en un CD e inicie desde el cdrom en la terminal que va a instalar.



Figura 1. Pantalla de instalación.

Instalación base de Debian

El proceso de la instalación se realiza en un modo consola. La terminal a instalar deberá de contar con dos placas de red, para mejor aislación entre la interconexión de nodos del cluster y la conexión a una red externa (internet). A continuación se detallan los pasos a pasos.

1. En la pantalla de Elegir Idioma, se selecciona "English".
2. En la elección de País, se selecciona "United States".
3. En la pantalla de seleccionar una configuración de teclado, se selecciona "American English".

-
4. El hardware es detectado, los controladores y componentes se cargan, la red es configurada.
 5. En la pantalla de configuración de red, se selecciona “eth0” como interfaz primaria de red.
 6. En la pantalla de configuración de red, se selecciona como hostname “Lenny”
 7. En la pantalla de configuración de red, se selecciona como nombre de dominio “clusterssi”.
 8. Discos de almacenamientos son detectados y el proceso de partición es iniciado.
 9. En la pantalla de partición de discos, se selecciona la opción “Manually edit partition table”
 10. La partición se procederá de la siguiente manera. 1.0 Gb para la partición **/boot** con sistema de archivos *ext3*, 0.5 Gb para la partición **/swap** y el espacio sobrante del disco se asigna a la partición **/** con sistema de archivos *ext3*.
 11. El sistema base de Debian ha sido instalado
 12. Se consultara acerca de instalar el cargador GRUB en el registro de inicio maestro, se responde “Yes”.

Configuraciones de Debian

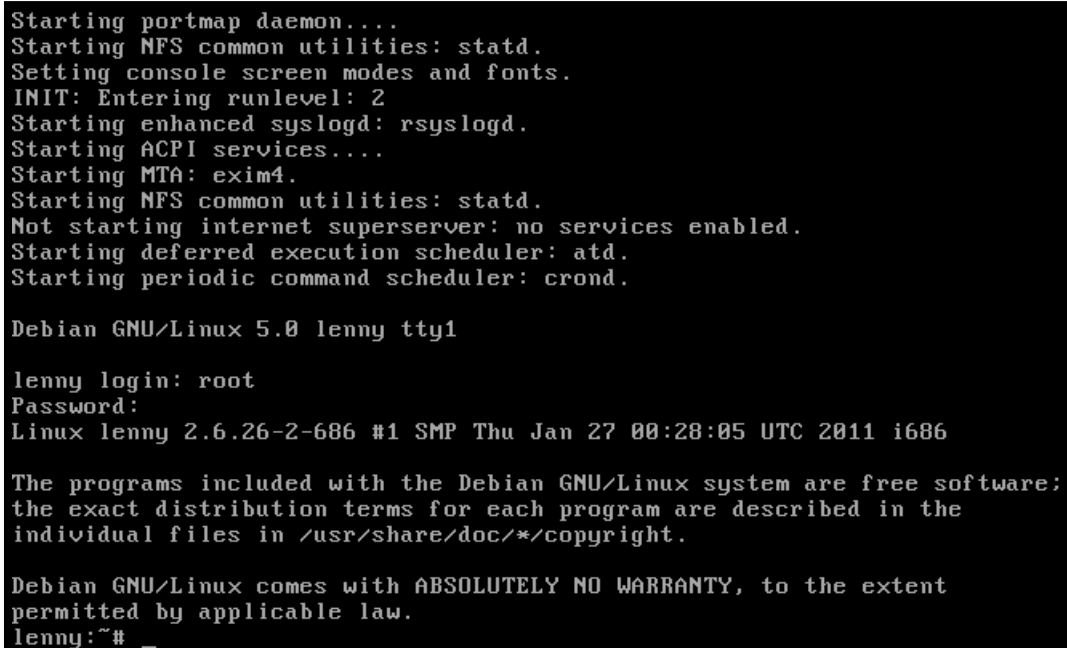
Una vez instalado el sistema base de Debian, se procede a realizar configuraciones básicas del sistema operativos.

1. En la pantalla de configuración de Huso Horario, se selecciona “No” cuando se consulta si el reloj interno del hardware esta configurado a GMT.
2. En la pantalla de configuración de Huso Horario, se selecciona “other” y luego

“South America”, “Paraguay” como huso horario.

3. Se ingresa y verifica una contraseña para el administrador “root”.
4. Se crea un usuario regular.
5. En la pantalla de configuración de paquetes APT, se selecciona “No” cuando se consulta si se desea examinar otro CD.
6. En la pantalla de configuración de paquetes APT, se selecciona “Yes” cuando se consulta acerca de agregar otro repositorio APT. Seleccione http, luego “United States” y luego “http://ftp.us.debian.org”. No es necesaria información para proxy HTTP.
7. Las configuraciones están hechas.

Una vez reiniciada la terminal la siguiente pantalla se visualizará el inicio de sesión de usuarios.



```
Starting portmap daemon....
Starting NFS common utilities: statd.
Setting console screen modes and fonts.
INIT: Entering runlevel: 2
Starting enhanced syslogd: rsyslogd.
Starting ACPI services....
Starting MTA: exim4.
Starting NFS common utilities: statd.
Not starting internet superserver: no services enabled.
Starting deferred execution scheduler: atd.
Starting periodic command scheduler: crond.

Debian GNU/Linux 5.0 lenny tty1

lenny login: root
Password:
Linux lenny 2.6.26-2-686 #1 SMP Thu Jan 27 00:28:05 UTC 2011 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
lenny:~# _
```

Figura 2. Inicio de sesión

Instalación del nodo maestro (master) o init

Posterior a la instalación del sistema operativo en la terminal que se desea usar como servidor o master (llamare master a partir de este momento) del cluster, se procede a las configuraciones para preparar al mismo. Se recomienda que se trabaje con únicamente la consola de Debian como entorno, ya que al instalar un entorno grafico, consumiría memoria ram, la cual puede ser utilizada de mejor manera en otros procesos del sistema.

Una vez iniciado el master con Debian Lenny instalado, se iniciara sesión como usuario administrador root, esto es para evitar escribir *sudo* a cada momento, luego de iniciada la sesión lo primero a realizar es la modificación del archivo “sources.list” de la siguiente manera.

```
1 nano /etc/apt/sources.list
```

Este se modificara para que quede de la siguiente forma:

```
1 deb http://ftp.us.debian.org/debian lenny main contrib
2 deb-src http://ftp.us.debian.org/debian lenny main contrib
3
4 deb http://security.debian.org/ lenny/updates main contrib
5 deb-src http://security.debian.org/ lenny/updates main contrib
6
7 deb http://deb.openssi.org/openssi 1.9.6-lenny-preview openssi extras
8 deb-src http://deb.openssi.org/openssi 1.9.6-lenny-preview openssi extras
```

Se crea el archivo “preferences” en la carpeta apt

```
1 nano /etc/apt/preferences
```

Con la finalidad de que al momento de realizar cualquier tipo de instalación de un paquete, el sistema tenga preferencia por aquellos que se encuentre en el servidor de OpenSSI, se agregan las siguientes líneas:

```
1 Package: *
2 Pin: origin deb.openssi.org
3 Pin-Priority: 1001
```

Por ultimo, con el fin de autorizar los paquetes no autenticados se crea el archivo 90auth para autenticación.

```
1 nano /etc/apt/apt.conf.d/90auth
```

Donde se ingresa la siguiente línea

```
1 APT::Get::AllowUnauthenticated "true";
```

Es importante que estos archivos contengan las líneas tal como se muestran expresadas para el correcta descarga e instalación de OpenSSI.

A continuación se ejecuta el comando *update* para actualizar repositorios y fuentes en Debian. Nota: si la salida de Internet es a través de un proxy se debe configurar primero, antes de continuar con los siguientes pasos.

```
1 apt-get update
```

Una vez realizada la operación anterior, se procede a instalar el paquete “initrd-tools”.

```
1 apt-get install initrd-tools
```

Se procede a desinstalar los siguientes paquetes:

```
1 apt-get remove nfs-common
2 apt-get remove libc6-amd64
```

Se procede a instalar el paquete “openbsd-inetd”

```
1 apt-get install openbsd-inetd
```

Se modificará el archivo de las interfaces de red:

```
1 nano /etc/network/interfaces
```

Se agrega al final del archivo lo siguiente

```
1 #Setting for the Cluster Interconnect
2 auto eth1
3 iface eth1 inet static
4 address 192.168.64.1
5 netmask 255.255.255.0
6 broadcast 192.168.64.255
```

Lo siguiente es agregar al modulo del NIC del grupo de interconexión al archivo modules, la tarjeta de red que el Master utilizara para comunicarse con los demás nodo, para obtener el NIC se usa la siguiente instrucción

```
1 ls -ld /sys/class/net/ethX/device/driver/module
```

La “X” es el numero de identificación de la tarjeta de red (para saber que numero tiene la tarjeta de red que deseas usar solo coloca “ifconfig” en la consola). En este caso, seria eth1. El NIC es el código que se encuentra al final de la línea después del ultimo “/” este código se debe agregar en la ultima línea en el siguiente archivo.

```
1 cat /etc/mkinitrd/modules
```

Una vez agregado el código, se procede a remover el kernel actual del sistema, para ello se debe hacer *modprobe* a los módulos necesarios para construir el *initrd*.

ADVERTENCIA: a partir de este punto el sistema no será iniciable, por lo tanto no se puede ni se debe apagar el master o cometer algún error, o sino será necesario comenzar de nuevo. Se realiza el modprobe de la siguiente forma:

```
1 modprobe loop
2 modprobe ext2
```

Ahora se debe instalar el kernel OpenSSI pre-compilado y se remueve el kernel actual, para ello se utiliza.

```
1 apt-get dist-upgrade
```

Nota I: si pregunta acerca abortar la remoción del kernel, se responde "no"

Nota II: si durante la instalación pregunta acerca "portmap" se contesta "I" (i latina)(sin las comillas).

Una vez instalado el nuevo kernel se verifica que el "udev" este en la versión 0.080-1

```
1 dpkg -l udev
```

Si no se encuentra en esa versión, debe de ejecutarse:

```
1 apt-get install udev
```

Una vez concluida esta parte, se debe configurar el fichero /etc/udev/rules.d empezando asi:

```
1 ls /etc/udev/rules.d
```

Una lista similar a esta aparecerá

```
1 50-udev.rules          70-persistent-net.rules
2 60-persistent-input.rules  75-cd-aliases-generator.rules
3 60-persistent-storage.rules  75-persistent-net-generator.rules
4 60-persistent-storage-tape.rules  80-drivers.rules
5 60-persistent-v4l.rules    91-permissions.rules
6 70-persistent-cd.rules     95-late.rules
```

Para solucionar se deben de ejecutar las siguientes líneas:

```
1 rm /etc/udev/rules.d/*
2 bash -c 'source /var/lib/dpkg/info/udev.postinst abort-deconfigure; create_rules_symlink'
3 ls -l /etc/udev/rules.d
```

Obteniendo el siguiente resultado.

```
1 total 0
2 lrwxrwxrwx 1 root root 20 2009-04-11 12:25 020_permissions.rules -> ../permissions.rules
3 lrwxrwxrwx 1 root root 19 2009-04-11 12:25 cd-aliases.rules -> ../cd-aliases.rules
4 lrwxrwxrwx 1 root root 13 2009-04-11 12:25 udev.rules -> ../udev.rules
5 lrwxrwxrwx 1 root root 19 2009-04-11 12:25 z20_persistent.rules -> ../persistent.rules
6 lrwxrwxrwx 1 root root 12 2009-04-11 12:25 z50_run.rules -> ../run.rules
7 lrwxrwxrwx 1 root root 16 2009-04-11 12:25 z55_hotplug.rules -> ../hotplug.rules
8 lrwxrwxrwx 1 root root 17 2009-04-11 12:25 z70_hotplugd.rules -> ../hotplugd.rules
```

El paso a continuación es instalar openssi en el sistema con la siguiente instrucción.

```
1 apt-get install openssi
```

En caso de que al finalizar la instalación, no se soliciten configuraciones del master, ejecutar el comando *ssi-create*.

```
Unpacking linux-ssi-image-2.6-686-smp (from ../linux-ssi-image-2.6-686-smp_102.ssi3_i386.deb) ...
This command requires a CI/OpenSSI kernel.
Welcome to OpenSSI clustering!

Let's configure the first node in your cluster.

Enter a node number (1-125) or (?) [1]: 1

Select a network interface for the cluster interconnect.

    Name      IP address      Netmask      Hardware address
    ----      -
1)  eth0      192.168.100.15   255.255.255.0  52:54:00:12:01:02
2)  eth1      192.168.64.1     255.255.255.0  52:54:00:12:01:00

Select (1-3), (R)escan or (?) [1]: 2

Enter a clustername or (?): openssi-lenny

Do you want to enable root failover (y/n/?) [n]: y

The following configuration has been entered:
Node number:      1
NIC for interconnect:  eth1
IP address:      192.168.64.1
Network hardware addr:  52:54:00:12:01:00
Local boot device:  UUID="967228e7-bdef-44cb-9f63-dd3d6dbd8e6c"
Clustername:      openssi-lenny
Root failover:      Yes
Potential initnode:  Yes
NFS support:      yes

(W)rite new configuration or (R)econfigure [W]: w
```

Estas son las configuraciones del OpenSSI para el master. Recuerde que la tarjeta de

red previamente configurada para la interconexión es la que se debe utilizar. El siguiente paso es instalar un kernel que corregirá el problema de inicialización del sistema operativo.

Para instalar un kernel de 32 bits se ejecuta

```
1 apt-get install linux-image-2.6.14-ssi-686-smp
```

O para instalar un kernel de 64 bits se ejecuta

```
1 apt-get install linux-image-2.6.12-ssi-amd64
```

Lo siguiente a hacer es reiniciar el master usando el comando *init 6*. Una vez reiniciado el master con el kernel OpenSSI, el master se vuelve un cluster de un nodo, por el momento, posteriormente se agregara mas nodos. El siguiente paso a realizar es modificar el archivo *dhcpd.proto*:

```
1 nano /etc/dhcp3/dhcpd.proto
```

Se agrega al final del archivo la siguiente línea:

```
1 next-server 192.168.64.1;
```

Luego se debe ejecutar el siguiente comando *mkdhcpd.conf*, se verifica que se haya generado el archivo *dhcpd.conf*

```
1 cat /etc/dhcp3/dhcpd.conf
```

Debería de mostrar algo similar a lo siguiente:

```
# /etc/dhcp3/dhcpd.conf
#
# Do _not_ edit this file!! It was automatically generated by mkdhcpd.conf.
#
# Section 1 (from /etc/dhcp3/dhcpd.proto)
next-server 192.168.64.1;
# Section 2 (from /etc/clustertab)
subnet 192.168.64.0 netmask 255.255.255.0 {
    host node1 {
        hardware ethernet 52:54:00:12:01:00;
        fixed-address 192.168.64.1;
    }
}
```

Se debe de reiniciar el proceso de dhcp3-server:

```
1 invoke-rc.d dhcp3-server restart
```

Ahora se debe asegurar de enlazar los directorios “/tftpboot” y “/var/lib/tftpboot” para ello se ejecuta:

```
1 ls -l /var/lib/tftpboot/
```

Lo cual debe de retornar en consola *total 0*. Luego se ejecuta:

```
1 mv /var/lib/tftpboot/ /var/lib/tftpboot.old
2 ln -s /tftpboot/ /var/lib/
3 ls /var/lib/tftpboot
```

Se obtiene:

```
combined initrd kernel pxelinux.0 pxelinux.cfg
```

A partir de este momento se pueden agregar nodos a la infraestructura OpenSSI.

Agregar nodos a la infraestructura.

Una vez instalado el sistema operativo en el nodo master, instalado el kernel openSSI en el mismo, se procede a agregar nodos para poder contar con una infraestructura Cluster. Para ello se deben realizar una serie de pasos y chequeos iniciando por la verificación si las tarjetas de red de los nodos esclavos no poseen soporte de booteo PXE. Se puede descargar un iso desde la siguiente url <http://rom-o-matic.net/gpxe/gpxe-1.0.1/contrib/rom-o-matic/> Debido a que usaremos PXE, no es necesaria una imagen Etherboot.

Si el NIC de la tarjeta de red de los nodos esclavos no se encuentra en el archivo “/etc/mkinitrd/modules” del master. Se debe editar este archivo

```
nano /etc/mkinitrd/modules
```

Agregar el nombre del driver al final del archivo y luego reconstruir el ramdisk para contener los módulos necesarios para el inicio, debido a que este mismo ramdisk es el que se envía después a los nodos esclavos la imagen del SO. Para generar de nuevo el ramdisk se ejecutan los siguientes comandos:

```
mkinitrd -o /boot/initrd.img-2.6.14-ssi-686-smp 2.6.14-ssi-686-smp  
ssk-ksync
```

Debido a que durante la instalación del sistema operativo al nodo master, se ingresó la opción noapic, es necesario realizar una modificación dentro del archivo /tftpboot/pxelinux.cfg/default.

```
nano /tftpboot/pxelinux.cfg/default
```

Se agrega “noapic” antes del initrd de la siguiente manera:

```
append noapic initrd=images/pmagic/initrd.gz root=/dev/ram0 init=/linuxrc ramdisk_size=100000
```

Una vez hechas estas configuraciones en el nodo master. Se procede a iniciar el nodo esclavo con la imagen ISO gPXE. En el nodo esclavo, se podrá apreciar como la imagen gPXE se inicia y busca algún servidor DHCP, presionando de manera combinada las teclas CTRL + B se accede a la consola, al ingresar el comando *autoboot* se obtendrá la dirección MAC de la tarjeta de red que se utilizará para la interconexión con el cluster.

En el nodo master se ejecuta el comando *ssi-addnode* de la siguiente manera:

```
ssi-addnode --hwaddress="MACADDRESS"
```

En donde MACADDRESS es la dirección MAC del nodo esclavo que se encuentra ejecutando gPXE. Al ejecutar el comando *ssi-addnode* se realizaran algunas preguntas acerca de cómo se desea configurar el nuevo nodo y que son los siguientes.

```
Do you want verbose prompts (y/n) [y]:
```

Se ingresa "Y"

```
Enter a node number (2-125) or (?) [2]: 2
```

Se ingresa "2"

```
Select (P)XE or (E)therboot as the network boot protocol for
this node. PXE is an Intel standard for network booting, and
many professional grade NICs have a PXE implementation pre-
installed on them. You can probably enable PXE with your BIOS
configuration tool. If you do not have a NIC with PXE, you can
make use of open-source project Etherboot, which lets you generate
a floppy or ROM image for a variety of different NICs.
```

```
Select (P)XE, (E)therboot or (?) [E]: P
```

Se ingresa "P"

```
The nodename should be unique in the cluster and it should
resolve to one of this node's IP addresses, so that client
NFS can work correctly. The nodename can resolve to either
the IP address you configured above for the interconnect,
or to one of external IP addresses that you might configure
below. The nodename can resolve to the IP address either in
DNS or in the cluster's /etc/hosts file.
```

```
The nodename is stored in /etc/nodename, which is a context-
dependent symlink (CDSL). In this case, the context is node
number, which means each node you add will have it's own view
of /etc/nodename containing its own hostname. To learn more
about CDSLs, please see /usr/share/doc/openssi/cdsl.
```

```
Enter a nodename or (?): lenny-node2
```

Se ingresa “Lenny-node2” o el nombre que se escoja para el nodo.

```
Do you want this node to be a root failover node? It _must_
have access to the root filesystem on a shared disk in order
to answer yes. If you answer yes, then this node can boot
first as a root node, so you should configure it with a local
boot device. This is done after this node joins the cluster
and is described in the installation instructions.
```

```
Enable root filesystem failover to this node (y/n/?) [n]: y
```

Si en el master se activo la tolerancia a fallos, se ingresa aquí “Y”. Una vez guardadas estas configuraciones, el nodo esclavo se unirá al cluster por red.

Para comprobar el estado del cluster ejecutar el siguiente comando

```
lenny:~# cluster -v
1:  UP
2:  UP
```

Para la tolerancia de falla en el cluster, se deben realizar los siguientes pasos.

```
lenny:~# onnode 2 /bin/sh
lenny-node2:~# mkfs -t ext3 /dev/sdal
mke2fs 1.40-WIP (14-Nov-2006)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
126976 inodes, 506016 blocks
25300 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67633152
62 block groups
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729, 204801, 221185, 401409

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 28 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

Luego se ejecuta el comando *ssi-chnode*

```
lenny-node2:~# ssi-chnode
Select a node number (1,2) [1]: 2

Select (P)XE, (E)therboot or (?) [P]: P
Enter a new boot device: /dev/hdal

The following configuration has been entered:
Node number:      2
IP address:       192.168.64.2
Network hardware addr: 52:54:00:12:02:00
Network boot protocol: PXE
Local boot device:  UUID="ea073072-54be-4b60-9fcf-88a871c69fdc"
Potential initnode: Yes

(W)rite new configuration, (R)econfigure, or (Q)uit without writing [W]:
The configuration changes have been saved.
Do you wish to configure another node (y/n) [n]:
Rebuilding the boot materials
/tmp/initrd.OzHutZ: 69.3%
Synchronizing network boot images: succeeded
Stopping DHCP server: dhcpd3.
Starting DHCP server: dhcpd3.
Synchronizing local boot devices
    syncing UUID="967228e7-bdef-44cb-9f63-dd3d6dbd8e6c" on node 1:  succeeded
    syncing UUID="ea073072-54be-4b60-9fcf-88a871c69fdc" on node 2:  succeeded

Node 2 has been updated.
```

A partir de este momento, el nodo puede iniciar desde su propio disco. Repita los pasos para agregar mas nodos al cluster OpenSSI.

Instalación de la herramienta visual de monitoreo.

Bench TPCC-UVA

Bench TPCh

Script para instanciar el Postgres para pruebas OLTP

Scripts utilitarios para pruebas OLAP

Descripción técnica de Equipos

Anexo de Formulario de Pruebas