UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

"CLUSTER TIPO BEOWULF PARA APLICACIONES DE VISUALIZACIÓN CIENTÍFICA Y REALIDAD VIRTUAL"

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A:

CÉSAR SANABRIA PINEDA

Director de Tesis: ING. OSCAR RENÉ VALDEZ CASILLAS

México 2005

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Agradecimientos

A mis padres:

Mario y Natalia:

Quiero agradecerles lo que ahora soy …

Gracias por darme la vida … por su amor, por las caricias,
por las sonrisas, por el aliento…

Gracias por enseñarme a crecer… por el ejemplo de honradez,
por las verdades, y los regaños …

Gracias por enseñarme a defender mis ideales y mis pensamientos,
Gracias por todos su consejos …

Gracias por estar a mi lado en el momento justo y el más anhelado….
cuando necesito sentir su apoyo…

Gracias con todo mi corazón, gracias por ser como son…gracias por ser mis papás.

A mis hermanas.

Claudia Edith y Cynthia Paloma

Gracias por ser uno de los apoyos más importantes en mi vida…

Por el tiempo compartido juntos…
jugando, estudiando, discutiendo y escuchándonos ….

Gracias por preocuparse y cuidarme como su hermano menor …

Gracias por enseñarme a mostrar carácter cuando todo está apunto de derrumbarse…

Gracias por mostrarme el significado de la hermandad y del apoyo incondicional.

A la Universidad Nacional Autónoma de México por los valores aprendidos y fomentados durante mi paso por sus instalaciones.

A todos aquellos universitarios que han luchado
por preservar los valores y carácter público de nuestra Universidad

Al Plan de Becarios de Cómputo de Alto Rendimiento de la DGSCA generación 2003 donde surgió la idea de este trabajo.

A Marce y Vero por ser mis compañeras durante la realización de este proyecto, gracias por el tiempo dedicado, la paciencia pero sobre todo por brindarme su amistad. Este trabajo también es de ustedes.

A todas aquellas personas especiales que conocí durante mi estancia en esta Universidad.

Al Ing. Oscar Valdez, mi asesor, por su guía, orientación y paciencia durante cada etapa de este proyecto.

Finalmente gracias a todas aquellas personas que me han brindado su apoyo para llevar a buen término este trabajo.

Contenido

1 Introducción 1

2 Conceptos básicos 5
2.1 Cómputo Paraleo 5
2.2 Clasificación de Flynn 6
2.2.1 SISD. Single Instruction, Single Data 6
2.2.2 SIMD. Single Instruction Multiple Data 7
2.2.3 MISD. Multiple Instruction, Single Data 8
2.2.4 MIMD. Multiple Instruction, Multiple Data. . . . 8
2.2.4.1 Memoria Compartida. 9
2.2.4.2 Memoria Distribuida. 10
2.2.5 Máquinas Masivamente Paralelas. 10
2.2.6 Procesamiento Distribuido 10
2.3 Cluster 11
2.4 Breve historia del Cluster Beowulf 12
2.5 Envío de Mensajes 14
2.5.1 PVM 14
2.5.2 MPI 15
2.5.3 MPICH 15
2.5.4 LAM 16

3 Consideraciones de diseño 17
3.1 Aplicaciones de Software Paralelas 17
3.2 Middleware 18
3.3 Sistema Operativo 18
3.4 Interconexión 19
3.4.1 Ethernet 21
3.4.2 SCI 21
3.4.3 Myrinet 22
3.4.4 Quadratics 22
3.4.5 Infiniband 22
3.5 Nodos 23
3.6 Ambiente 25
3.7 Arquitectura del cluster Zion 25

3.8 Sistema de archivos 27

4 Instalación 31
4.1 Instalación del nodo maestro 32
4.2 Personalización del kernel 36
4.3 Tarjeta gráfica 41
4.4 Instalación de bibliotecas de envío de mensajes MPICH y LAM . . 42
4.5 Instalación de los nodos internos 44

5 Administración 45
5.1 Administración local 45
5.1.1 Arranque y apagado del sistema. 45
5.1.2 Administración de usuarios y grupos 45
5.1.2.1 Alta de usuarios. 46
5.1.2.2 Baja de usuarios 47
5.1.3 Administración de recursos del sistema . . . 47
5.1.4 Administración del sistema de archivos 54
5.1.5 Cuotas del sistema 54
5.1.6 Backup y restauración del sistema 56
5.1.7 Automatización de tareas rutinarias 56
5.1.8 Seguridad del sistema 57
5.1.9 Monitoreo del sistema 57
5.1.10 Ajuste y personalización. 57
5.2 Administración de red 58

6 Seguridad 65
6.1 Buenas prácticas. 66
6.2 Seguridad local 67
6.3 Seguridad en red 70
6.3.1 Protección mediante filtrado. TCPwrappers . . . 70
6.3.2 Firewall 71

6.4 Seguridad en la Administración 73
6.4.1 Protección de la cuenta root 73

7 Monitoreo 75
7.1 Análisis de bitácoras (logs) 75

7.2 Monitoreo del desempeño del sistema (Ganglia) 79
7.2.1 Ganglia 79
7.2.2 Instalación de Ganglia 82
7.2.3 Configuración 82
7.2.3.1 Configuración del demonio GMOND . . . 82
7.2.3.2 Configuración del demonio GMETAD . . . 83

8 Aplicaciones gráficas instaladas y ejemplos 86
8.1 vr++ 0.7.1 1 86
8.2 Performer 87
8.3 Vrjuggler 88
8.4 Netjuggler 88
8.5 VTK 90
8.6 Paraview 91
8.7 POVRAY 92
8.8 OpenDX 93
8.9 OpenScenegraph-0.9.6-2 94
8.10 VrNAv2-5.8 94
8.11 Chromium 95
8.12 OpenSG 95
8.13 Syzygy 97
8.14 OpenInventor 97
8.15 ITK 99

9 Conclusiones 102

Referencias 105
Anexo A
Anexo B
Anexo C

Capítulo 1. Introducción

En muchas ramas de la ciencia, la complejidad de los problemas que se estudian requiere contar con acceso a una computadora capaz de desarrollar varios miles de millones de operaciones por segundo. Las supercomputadoras tradicionales emplean procesamiento en paralelo; contienen matrices de microprocesadores muy rápidos que trabajan en sincronía para resolver problemas complejos. Los fabricantes de computadoras como Cray, IBM, Silicon Graphics, HP, NEC, entre otros, producen modelos por diseño especial que cuestan varios millones de dólares, precios que van más allá de los presupuestos de inversión de la gran mayoría de los grupos de investigación.

En los últimos años, el personal académico de diversas universidades y centros de investigación, entre ellos la UNAM, se ha dado a la tarea de aprender a construir sus propias “supercomputadoras” conectando computadoras personales y desarrollando software para enfrentar tales problemas.

La Universidad Nacional Autónoma de México es la institución académica que con la llegada de la primer supercomputadora CRAY YMP/4 en 1991, a través de la DGSCA, ha ofrecido los recursos y mecanismos para procesar grandes volúmenes de información y dar solución a problemas de cómputo considerados de gran reto.

Así mismo, y a pesar de los recursos de supercómputo con que cuenta la UNAM, es necesario crear y adaptar sistemas para el tratamiento de información y aplicaciones muy específicas, ofreciéndolos como una alternativa a áreas especializadas como son Visualización Científica y Realidad Virtual. Como se mencionó anteriormente, disponer de este tipo de infraestructura es por demás costoso, por ello se considera al cluster una de las opciones más viables ya que proporciona recursos computacionales requeridos en ambientes de alto desempeño y gran disponibilidad, permitiendo a través del cómputo paralelo y distribuido resolver eficientemente problemas de cómputo a un costo bastante aceptable.

En este contexto, y con la reciente introducción de tarjetas gráficas profesionales a equipos comunes de cómputo, se ha logrado utilizar a los clusters en aplicaciones de visualización científica y realidad virtual inmersiva. Dentro de las características que debe cumplir este tipo de clusters están: la opción de ejecución de sesiones gráficas en cada nodo del cluster de manera simultánea, además de la sincronización de las imágenes generadas y desplegadas por los diferentes componentes del arreglo, de modo que aparezcan como una sola.

La construcción del cluster para aplicaciones de visualización interactiva y distribuida de datos científicos y la generación eficiente de ambientes virtuales inmersivos formó parte del proyecto final del Plan de Becarios de Cómputo de Alto Rendimiento de la Dirección General de Servicios de Cómputo Académico (DGSCA) de la UNAM, generación 2003, en el área de especialización de administración de sistemas unix y presentado el día 29 de Octubre de 2004 dentro del coloquio del plan de becarios en el auditorio de la DGSCA.

El objetivo de esta tesis es la implementación de un cluster tipo Beowulf para aplicaciones de visualización científica y realidad virtual, documentando

paso a paso los procesos de diseño, intalación, configuración y administración de éste.

En el Capítulo 2 se da una breve introducción a los conceptos como cómputo paralelo, la clasificación de computadoras, también una introducción al concepto de cluster de computadoras, de un cluster tipo beowulf así como el de librerías de envío de mensajes parte esencial de la construcción y configuración de un cluster.

En el Capítulo 3 se tratan los aspectos más importantes que se tomaron en cuenta para la construcción del cluster tanto de hardware como de software. Así mismo, se explica brevemente cual es la arquitectura actual y esquema del sistema de archivos.

El Capítulo 4 abarca todo el proceso de instalación del cluster, desde el sistema operativo hasta las aplicaciones de visualización y realidad virtual así como también la creación de una imagen de kernel que será común para el resto de los nodos que conforman el cluster.

En el Capítulo 5 se muestra el proceso de administración del cluster, tanto de usuarios, red y del sistema en su conjunto, aspectos como el alta y baja de usuarios, la generación de grupos, el esquema de cuotas, la configuración de la red interna, del sistema de archivos de red (NFS) y algunas herramientas desarrolladas para facilitar la administración.

En el Capítulo 6 se desglosan todos los mecanismos de seguridad implementados en el cluster siendo la parte principal la creación de su firewall personal que protege al cluster de ataques externos.

Durante el Capítulo 7 se trata el aspecto de monitoreo del sistema entre los que se encuentra el análisis y monitoreo oportuno de bitácoras del sistema

empleando logwatch. Así mismo, se muestra la instalación y configuración de la utilería llamada ganglia que permite el monitoreo de la carga del sistema, de cpu, red, etc.

En el Capítulo 8 de muestra el proceso de instalación de las aplicaciones gráficas de visualización y realidad virutal instaladas en el cluster así como algunos ejemplos de su ejecución.

En el Capítulo 9 se describen las conclusiones a las que se llegó durante la elaboración del presente trabajo.

Capítulo 2. Conceptos básicos

2.1 Cómputo Paralelo.

Tradicionalmente, las computadoras convencionales procesan cada instrucción de forma secuencial, es decir, los problemas son resueltos mediante una serie de instrucciones ejecutadas una tras otra por el CPU, por lo que sólo una instrucción puede ser ejecutada a la vez. Sin embargo en problemas computacionales que requieren mayor cantidad de procesamiento, el tiempo de ejecución crece exponencialmente por lo que el cómputo paralelo se ha convertido en la manera de solucionar éste tipo de problemas conocidos como de gran reto realizando un procesamiento concurrente de datos para conseguir un menor tiempo de ejecución.

El cómputo paralelo no es más que el uso simultáneo de múltiples recursos de cómputo para resolver un problema computacional. Dichos recursos pueden ser una computadora con varios procesadores, un número arbitrario de computadoras conectadas a una red o una combinación de ambas [10]. Emplea un grupo de técnicas que se usan para proporcionar tareas simultáneas de procesamiento de datos con el fin de aumentar la velocidad computacional de un sistema de computadoras.

Para fines prácticos, es necesario tener en mente que la paralelización es un factor básico para tener un alto desempeño en los equipos de cómputo. Así el problema computacional a resolver debe mostrar características como la habilidad de fragmentarse en partes que puedan ser resueltas simultáneamente, ejecutar múltiples operaciones a la vez y debe reducir al mínimo el tiempo total de cómputo distribuyendo la carga de trabajo entre los procesadores disponibles.

Existen varias maneras de clasificar el cómputo paralelo. Puede considerarse a partir de la organización interna de los procesadores, desde la estructura de interconexión entre los procesadores o desde del flujo de información a través del sistema. Una clasificación propuesta por M. J. Flynn considera la organización de un sistema de computadoras mediante la cantidad de instrucciones y unidades de datos que se manipulan en forma simultánea.

2.2 Clasificación de Flynn

La clasificación de Flynn está basada en la forma en que las computadoras manipulan los datos y las instrucciones de acuerdo a su arquitectura. Es una clasificación útil por su simplicidad, sin embargo ha recibido críticas ya que las máquinas actuales se han diversificado abarcando varias categorías y/o restringiendo algunas de las categorías [7].

Las categorías son:

- **SISD**. Single Instruction, Single Data (Una Instrucción, Un Dato)
- **SIMD**. Single Instruction Multiple Data (Una Instrucción, Múltiples Datos).
- **MISD**. Multiple Instruction, Single Data (Múltiples Instrucciones, Un Dato)
- **MIMD**. Multiple Instruction, Multiple Data. (Múltiples Instrucciones, Múltiples Datos)

2.2.1 SISD. Single Instruction, Single Data (Una Instrucción, Un Dato).

Este tipo de máquina se compone de un CPU que procesa las instrucciones de manera serial. Las supercomputadoras actuales tienen más de un CPU, pero si éstos trabajan de manera independiente con las instrucciones y con los datos, hablamos de una máquina SISD. Ejemplos: Estaciones de trabajo DEC, HP, SUN.

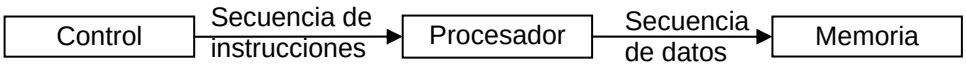

Figura 2.1. Modelo SISD (Single Instruction Single Data) [10].

Para procesar la suma de N números $a_1, a_2, \ldots a_N$, el procesador necesita acceder a memoria N veces consecutivas (para recibir un número). También son ejecutadas en secuencia N-1 adiciones. Es decir los algoritmos para las computadoras SISD no contienen ningún paralelismo, éstas están constituidas de un procesador.

2.2.2 SIMD. Single Instruction Multiple Data (Una Instrucción, Múltiple Datos).

Estas máquinas contienen varios CPU's los cuales trabajan en paralelo ejecutando las mismas instrucciones sobre conjuntos diferentes de datos. Ejemplos: CPP DAP Gamma, Alenia Quadrics.

Una subclase de las máquinas SIMD son las de procesadores vectoriales, éstas trabajan sobre un conjunto de datos de manera paralela, ejecutando la misma instrucción sobre cada conjunto de datos (que se denomina vector) en uno o varios CPUs. Ejemplos Hitachi S3600, Cray YMP.

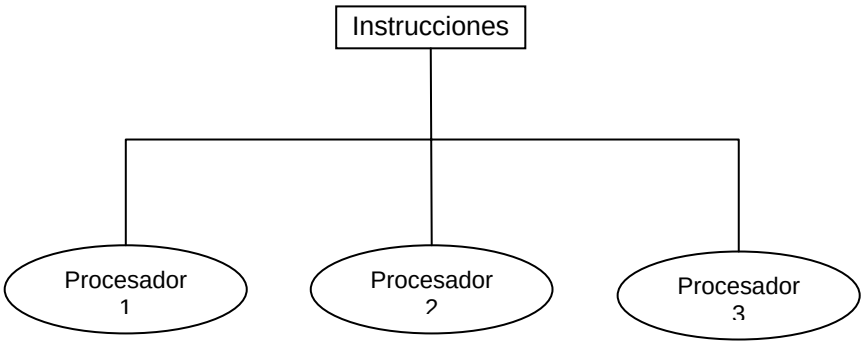

Figura 2.2. Modelo SIMD (Single Instruction Multiple Data) [10].

Aquí hay N secuencias de datos, una por procesador, así que diferentes datos pueden ser utilizados en cada procesador. Los procesadores operan sincronizadamente y un reloj global se utiliza para asegurar esta operación. Es decir, en cada paso todos lo procesadores ejecutan la misma instrucción, cada uno en diferente dato.

2.2.3 MISD. Multiple Instruction, Single Data (Múltiples Instrucciones, Un Dato)

Esta clase de máquina teóricamente procesa múltiples instrucciones sobre un mismo conjunto de datos. Sin embargo, no existe ninguna máquina que corresponda a dicha clasificación.

2.2.4 MIMD. Multiple Instruction, Multiple Data (Múltiples Instrucciones, Múltiples Datos)

Estas máquinas contienen una mayor cantidad de CPU's, los cuales al ejecutar un proceso pueden repartir tanto los datos como las instrucciones en los diferentes procesadores. Difiere de las SIMD en que las diferentes instrucciones pueden estar relacionadas entre si. Ejemplos: NEC SX'5, Cray T3E.

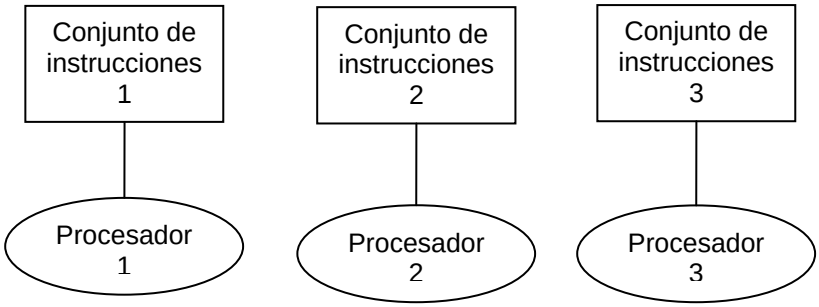

Figura 2.3. Modelo MIMD (Multiple Instruction Multiple Data)[10].

Se tienen N procesadores, N secuencias de instrucciones y N secuencias de datos. Cada procesador opera bajo el control de una secuencia de instrucciones, ejecutada por su propia unidad de control, es decir cada procesador es capaz de ejecutar su propio programa con diferentes datos. Esto significa que los procesadores operan asíncronamente, o en términos simples, pueden estar haciendo diferentes cosas en diferentes datos al mismo tiempo.

Las máquinas MIMD (de igual manera las SIMD) pueden dividirse en dos tipos de acuerdo al manejo de la memoria:

2.2.4.1 Memoria Compartida (SM)

Múltiples CPU's comparten el mismo espacio físico de memoria. Todos los procesadores accesan a la memoria de la misma forma.

Tiene un número de procesadores pequeño ya que son difícilmente escalables, esto es la capacidad que tiene para incrementar el número de procesadores ya que al aumentarlos el tráfico en el bus que conecta a los procesadores y a la memoria compartida aumenta y por lo tanto baja su eficiencia. Ejemplos: Nodos Cray V1, NEC SX-5, Compaq ServerSC.

2.2.4.2 Memoria Distribuida (DM)

Cada CPU tiene conectada su propia memoria (este arreglo se le conoce como nodo), sin embargo, están conectados por algún tipo de red, de manera que pueden intercambiar datos entre ellos, por medio del envío de mensajes. Su escalabilidad es mayor que en las de memoria compartida, sin embargo, la velocidad de comunicación entre los procesadores es menor (dependiente de la eficiencia en el envío de mensajes) y éste parámetro es muy importante en estas máquinas. Ejemplos: SGI Origin.

2.2.5 Máquinas Masivamente Paralelas

Dentro de esta categoría se tiene las máquinas que dada la gran cantidad de procesadores que tienen es imposible que tengan memoria compartida. Sin embargo, el acceder a una memoria distribuida también causa problemas principalmente para el usuario final. Por ello, estas máquinas ven a todo el sistema como uno solo, incluyendo la memoria, con tecnologías como la cc-NUMA (Coherence Cache-Non Uniform Access Memory). Ejemplos: SGI Origin, HP.

Algunas máquinas utilizan ambos esquemas (SM MIMD y DM MIMD) ya que dentro de un nodo de varios procesadores utilizan memoria compartida, pero con otros nodos tienen memoria distribuida. Ejemplos: NEC SX-5, Compaq ServerSC, SGI Origin.

2.2.6 Procesamiento Distribuido.

Otra clasificación que podría entrar dentro del DM-MIMD es el de procesamiento distribuido, pero en vez de tener una máquina con muchos

procesadores integrados, se tiene un conjunto de máquinas, servidores o estaciones de trabajo conectadas por una red de gran velocidad (Gigabit) Ethernet, FDDI, ATM u otra y que trabajan de manera conjunta.

La principal diferencia entre estas máquinas y las MIMD es la velocidad de comunicación entre procesadores, que en las primeras es mucho menor que en las MIMD. La comunicación debe hacerse con envío de mensajes (MPI, PVM).

Los clusters tema que nos atañe en el presente trabajo, son una forma de máquina con memoria distribuida y por consiguiente MIMD para cómputo paralelo. Son aplicados para propósitos específicos como alta disponibilidad, capacidad y desempeño y por lo tanto requieren software con diferentes funciones, puesta a punto de aplicaciones, herramientas y librerías.

2.3 Cluster

Existen una gran variedad de definiciones de lo que es un cluster, algunas muy similares otras un poco más profundas, sin embargo, tratando de encontrar una definición sencilla y que explique de manera general el concepto de un cluster tenemos:

Un **cluster** es un tipo de arquitectura paralela distribuida que consiste de un conjunto de computadores independientes interconectados operando de forma conjunta como un único recurso computacional, sin embargo, cada nodo puede utilizarse de forma independiente o separada.

Por definición, un cluster debe consistir de al menos dos nodos, un maestro y un esclavo. El nodo maestro es la computadora con la que los usuarios interactuarán. El nodo maestro puede participar en el cómputo al igual que los

demás nodos pero no es requerido e incluso recomendado en clusters grandes. Los nodos esclavos son sólo eso, responden a solicitudes del nodo maestro y, en general, hacer la mayor parte del procesamiento de datos.

Partiendo de la definición del cluster entramos a otra definición más particular de un cluster, ésta es la de un cluster Beowulf. La definición más aceptada de un cluster Beowulf es: un cluster de computadoras interconectadas a una red con las siguientes características las cuales también definen aspectos a considerar en el diseño de un Beowulf [36]

- Los nodos del cluster son dedicados al Beowulf y no sirven para otros propósitos.
- La red en la que residen los nodos es dedicada y no sirve para otros propósitos.
- La parte principal de un Beowulf y que lo distingue de otros cluster comerciales es que está construido con equipo fácil de encontrar y barato.
- La red también está construida con equipo económico y fácil de encontrar.
- Todos los nodos ejecutan software libre
- El cluster resultante es dedicado para Cómputo de Alto Desempeño (HPC por sus siglas en inglés).

2.4 Breve historia del Cluster Beowulf

El desarrollo del primer sistema Beowulf comenzó en el centro de la NASA Goddard Space Flight Center en 1994. El beowulf construido estaba formado por 16 nodos compuestos por procesadores 486DX a 100MHz, con 16MB de RAM y 2 tarjetas de red Ethernet a 10MB/s. En este sistema se usan 2 tarjetas de red en configuración de channel bonding (combinación de canales de datos para

formar una canal con mayor capacidad) para compensar la mayor velocidad del CPU con respecto a la red.

El siguiente cluster construido en la NASA estaba formado por 16 nodos compuestos de procesadores Pentium a 100MHz, 32MB en RAM y 2/3 tarjetas de red Fast Ethernet. Para compensar el aumento de velocidad del procesador se aumenta también la velocidad de la red a 1000Mb/s [3].

Después del éxito obtenido por esta clase de clusters se han instalado multitud de clusters Beowulf, tanto en instituciones académicas, como gubernamentales y de la industria. Además de redes Ethernet a 10 Mb/s y 100Mb/s se ha pasado a usar Gigabit Ethernet. En la UNAM, a fines de los 90's el Departamento de Supercómputo de la DGSCA inició la evaluación de tecnologías emergentes para construir máquinas para procesamiento paralelo también conocidas como clusters.

En el año 2001 se instaló y configuró un cluster experimental con la finalidad de preparar a estudiantes y personal en las diversas actividades que involucran el uso eficiente de recursos en un cluster de uso no-dedicado. Este primer cluster de prueba fue construido de computadoras personales conectadas por medio de un switch Fast-Ethernet.

En febrero de 2002 se adquirió el primer cluster de producción/experimentación; con esta adquisición, el Departamento de Supercómputo se integra a las diferentes instituciones y/o dependencias que están usando clusters para resolver problemas del cómputo científico [6].

2.5 Envío de mensajes

Cuando se diseña un cluster Beowulf, uno de los componentes más importantes es el conjunto de librerías que permiten dividir los datos y mandarlos a múltiples nodos consiguiendo que varias computadoras funcionen como un solo recurso de cómputo sobre una red. Existen dos estándares principales para el envío de mensajes PVM (Parallel Virtual Machine) y MPI (Message Passing Interface). De ésta última librería existen dos implementaciones (MPICH y LAM).

2.5.1 PVM

PVM es una biblioteca de envío de mensajes que hace uso de los recursos existentes en una red, sin importar que ésta sea homogénea o heterogénea, para ser usados como un solo recurso de cómputo, consiguiendo con ello un sistema paralelo de bajo costo. Esta biblioteca se desarrolló basándose en el concepto de máquina virtual, es decir, de una colección de recursos computacionales manejados como una computadora paralela. Es altamente portátil y está implementada generalmente sobre sockets. De esta forma los problemas que impliquen cómputo a gran escala, pueden ser resueltos de manera más efectiva a través del uso de la capacidad de memoria de varias computadoras.

Entre las características principales de PVM se encuentran que permite manipular grupos de computadoras (hosts) para conformar la máquina virtual, da acceso transparente al hardware de los nodos, esta basada en procesos y cuenta con un modelo de envío de mensajes explícito, maneja un soporte heterogéneo para arquitecturas de hardware, redes y aplicaciones, cuenta con soporte para Multiprocesadores.

2.5.2 MPI (Message Passing Interface)

La biblioteca de envío de mensajes MPI es un estándar soportado por la mayoría de los proveedores de hardware y ofrece más funcionalidad que PVM. La especificación de MPI la realiza un comité de estándares, mientras que la especificación de PVM la realizan sus propietarios. Dentro de las ventajas de MPI se encuentra la disponibilidad de varios modos de comunicación, los cuales permiten al programador el uso de buffers para el envío rápido de mensajes cortos, la sincronización de procesos o el traslape de procesos de cómputo con procesos de comunicación. Esto último reduce el tiempo de ejecución de un programa paralelo, no obstante, tiene la desventaja de que el programador debe ser más cuidadoso para evitar la corrupción de mensajes. Dos de las principales distribuciones de MPI son: MPICH y LAM/MPI .

2.5.3 MPICH

MPICH está diseñado como una implementación portátil y completa del estándar MPI 1.1, como LAM, permite que programas MPI puedan ser ejecutados dentro de una computadora o en un cluster de computadoras; contiene un ambiente de programación para trabajar con programas MPI, bibliotecas para el estudio de desempeño de los programas y un interfaz X para todas las herramientas. Su nombre se deriva de MPI y Chameleon, ya que MPICH usaba el sistema portable de envío de mensajes Chamelelon, posee herramientas para construir, correr, depurar y analizar programas MPI. Utiliza cuatro comandos para compilar programas en C (mpicc), C++ (mpiCC), Fortran 77 (mpif77) y Fortran 90 (mpif90).

2.5.4 LAM (Local Area Multicomputer)

LAM cuenta con una implementación completa del estándar MPI (con excepción que LAM no soporta cancelación de envíos); lo que trae como resultado, que el código fuente de cualquier aplicación desarrollada bajo LAM sea portable a cualquier otra implementación de MPI. La biblioteca también incluye facilidades para el control de la ejecución y ayudas para desarrollo y depuración de programas.

Las principales características de LAM son la implementación completa del estándar MPI-1 y especificaciones de MPI-2, soporta redes con estaciones de trabajo heterogéneas, contiene herramientas para depurar procesos y mensajes, los nodos son dinámicos, implementa tolerancia a fallos, comunicación rápida cliente-a-cliente, la cola de mensajes y los procesos bloqueados pueden ser examinados en tiempo de ejecución, la eliminación no planificada de un nodo, debido a la falla de una máquina o cualquier otra falla es detectada por LAM, quien se encarga de avisar a todos los procesos, a través de una señal asíncrona y finalmente tanto los nodos de LAM como los procesos MPI pueden ser agregados y removidos dinámicamente a tiempo de ejecución, en lo que se denomina un ambiente dinámico de recursos.

Capítulo 3. Consideraciones de diseño

Al diseñar y construir un Beowulf, es necesario tener en cuenta diversos factores que pueden llegar a afectar el desempeño del sistema algunos de ellos y que son explicados a continuación son considerados como mínimos para la construcción de un beowulf [2].

En términos generales, dichos factores son:

- Aplicaciones de software paralelas.
- Middleware
- Imagen de sistema operativo corriendo en cada nodo
- Tecnología de interconexión (Ethernet, Myrinet)
- Nodos (uni, dual o multi procesador)
- Ambiente físico.

3.1 Aplicaciones de software paralelas.

Estas aplicaciones son desarrolladas en ambiente paralelos que utilizan envío de mensajes para sincronizar los datos y resultados de una operación. Dichas aplicaciones son creadas empleando compiladores, herramientas y

librerías de programación optimizadas para la arquitectura del hardware a utilizar en los nodos del cluster.

3.2 Middleware

Por middleware entendemos a los componentes necesarios para operar y administrar el cluster como las librerías de envío de mensajes (MPI, PVM) (vistas en el capítulo anterior), herramientas administrativas, etc.

3.3 Sistema operativo

Actualmente, uno de los sistemas operativo más utilizados para la construcción de Clusters es Linux. Este sistema funciona en una variedad de procesadores entre los que se incluyen la familia x86 y alpha. La disponibilidad de su código fuente, ha permitido que se realicen modificaciones sobre él para adaptarlo a las necesidades de los clusters.

Linux y UNIX poseen características muy similares como son: multiplataforma, multitarea, multiusuario, protección de la memoria, memoria virtual, capacidades de red superiores, independencia de dispositivos, flexibilidad, herramientas y procedimientos de seguridad, soporte a consolas virtuales, etc. Sin embargo, Linux posee algunas características únicas:

- Licencia GNU GPL (General Public License); esta licencia promueve el uso, distribución y desarrollo de software libre.
- Distribuciones: una distribución es el conjunto del kernel (el cual es común en todas ellas) y aplicaciones adicionales. Algunas de las más conocidas son: RedHat, Debian, Mandrivia, SuSE, Gentoo y Slackware entre otras.

- Acceso al código fuente: El sistema es visto internamente por programadores expertos lo que permite la detección, y estudio detallado de las deficiencias y puntos vulnerables, agilizando la libración de soluciones a dichos problemas.
- Adaptabilidad tanto del kernel como de la gran mayoría de las aplicaciones, se tiene acceso total al código fuente con lo cual se puede hacer el sistema justo a la medida y necesidades.

La distribución a elegir depende en gran medida del administrador del sistema, es decir, de la comodidad que encuentre éste para su instalación, administración y mantenimiento. En el caso específico del cluster Zion, la distribución de Linux empleada es Debian. Debian es un proyecto completamente OpenSource, desarrollado y soportado por cientos de programadores de todo el mundo y considerada una de las distribuciones más seguras, estable y robusta pero también complicada en su instalación y administración. El sistema de paquetes Debian (.deb) es uno de los más avanzados sistemas de empaquetamiento, garantiza que los paquetes de su rama estable han sido ampliamente probados y son aptos para trabajar en un sistema en producción.

3.4 Interconexión

El tipo de interconexión es crucial en el desempeño de aplicaciones paralelas. La tecnología de interconexión define por sí misma el protocolo de comunicación. En un Cluster, el hardware debe conocer las necesidades del software incluyendo el tipo red a utilizar. Las tecnologías de red más típicas utilizadas en los clusters son [11]:

- 100BASE Fast Ethernet utilizando cable o fibra.
- 1000BASE Gigabit Ethernet cable o fibra.

- Scalable Coherent Interface. SCI
- Myricom, Myrinet
- Quadratics
- Arquitectura Infiniband

La velocidad y el desempeño de la red afecta directamente al desempeño del cluster. Se deben considerar dos características principales: la latencia entre los paquetes y el máximo ancho de banda posible en la red. La latencia es el tiempo de espera necesario para que un paquete de información viaje desde la fuente hasta su destino. El ancho de banda es el volumen de información que puede circular por un medio físico de comunicación de datos es decir, la capacidad máxima de transmisión y se mide en bits por segundo (bps). La latencia y el ancho de banda, juntos, definen la capacidad de velocidad de una red.

Protocolos con una latencia alta como TCP/IP y un ancho de banda limitado pueden afectar considerablemente el desempeño de un cluster por lo que para aplicaciones críticas se recomienda contar con una interconexión con poca latencia y un alto ancho de banda. La tabla siguiente muestra las características de desempeño entre diferentes tecnologías de red:

Interconexión	**Latencia μseg**	**Ancho de banda**
100 BASE	80	12 Mbps
1000 BASE	50	125Mbps
Myrinet/32bit	10	80Mbps
Myrinet/64-bit	10	160Mbps
SCI	5	200Mbps
Infiniband	-	2.5,10,30 Gbps

Tabla 3.1. Tecnologías de red para cluster.
Fuente: CLRC, Caresbury Lab, Nov 28, 2001, Steve Andrews

3.4.1 Ethernet

Ethernet ha sido la tecnología de red local usada más ampliamente. El estándar Ethernet trasmite a 10Mbps y ha sido usado exitosamente en el cómputo con clusters de alto rendimiento. Sin embargo, este ancho de banda ya no es suficiente para usarse en cómputo para clusters ya que no está balanceado con el poder de procesamiento de las computadoras disponibles ahora. La red Fast Ethernet provee un ancho de banda de 100Mbps a un costo muy bajo, y la Gigabit Ethernet provee mayor ancho de banda por un costo un tanto mayor. La red Gigabit Ethernet transmite señales a 1Gbps (de ahí el nombre Gigabit Ethernet), pero dada la codificación requerida a altas velocidades tiene una razón teórica de envío de datos de usuario de 800Mbps, o cerca de 100Mbytes/s. Ya que el protocolo se sobrecarga cuando se usa MPI, la razón medida para datos de usuario de la Gigabit Ethernet es aproximadamente 50 Mbytes/s con las computadoras de nueva generación. Esta razón se incrementa al incrementarse la velocidad del CPU [11].

3.4.2 SCI (Scalable Coherent Interface)

SCI fue la primer tecnología de interconexión desarrollada como un estándar específico para propósitos de cómputo con clusters. Para superar el rendimiento de los sistemas basados en bus, es una arquitectura punto a punto con tamaños de paquete pequeños y transacciones separadas mientras mantiene la impresión de una funcionalidad por bus en las capas externas. El ancho de banda de la implementación actual de las conexiones SCI se encuentra entre 400 y 1000 Mbytes/s y las latencias típicas de acceso remoto son de poco más de 2 micro segundos.

3.4.3 Myrinet

Myrinet es una red full dúplex a 1.28Gbps diseñada por Myricom. Fue usada en el cluster de estaciones de trabajo de Berkeley y muchos otros clusters académicos. Myrinet tiene un procesador integrado programable que permite experimentar con los protocolos de comunicación.

3.4.4 Quadratics. QsNet

Es una interconexión de gran ancho de banda y baja latencia para computadoras multiproceso. Provee soporte de programación paralela vía MPI, procesos de memoria compartida y TCP/IP. Soporta un protocolo verdadero de cero copia (de memoria virtual a virtual) y tiene un excelente desempeño. Aunque el costo actual por QsNet es algo mayor que otras redes para clusters de alto rendimiento, es posible que baje conforme el producto madure, haciendo a QsNet un producto de interconexión de clusters muy competitivo.

3.4.5 Infiniband

Un gran consorcio de miembros de la industria, incluyendo Compaq, Dell, Hewlett-Packard, IBM, Intel, Microsoft y Sun Microsystems, soporta el estándar InfiniBand. La arquitectura de InfiniBand reemplaza el bus compartido que es el estándar para Entrada/Salida en las computadoras actuales fabricadas con un puerto serial de alta velocidad, basada en canales, para envío de mensajes, escalable y para switches. Un solo conector de InfiniBand opera a 2.5Gbps, punto a punto en una sola dirección. Los datos son enviados en paquetes, y hay seis tipos de métodos de transferencia, incluyendo:

- Conexiones confiables y no confiables

- Datagramas confiables y no confiables
- Conexiones de envío múltiple
- Paquetes simples

3.5 Nodos

La arquitectura de los nodos incluye la arquitectura del procesador, el número de procesadores, memoria, espacio en disco etc. Para maximizar los beneficios de un cluster, se debe utilizar el hardware correcto. Generalmente, se dice que para un óptimo desempeño, todos los nodos excepto el maestro deben tener las mismas especificaciones de hardware. Esto, debido a que si un nodo toma más tiempo en hacer su trabajo, puede hacer lento al cluster entero, es decir, el resto de los nodos deberán detener lo que estén haciendo y esperar al nodo para nivelarse. Esto no sucede con frecuencia, sin embargo, es una consideración a tomar en cuenta. Al tener hardware idéntico en los nodos esclavos, se simplifica el proceso de configuración de los nodos al no tener que hacerla individualmente.

Para el nodo maestro, los aspectos a considerar son:

- **Velocidad del Procesador.**

Es especialmente crítico si el nodo maestro participa en el cómputo de datos. El nodo maestro deberá ejecutar mucho más tareas que los nodos esclavos así un procesador más rápido será requerido para evitar que se retrase respecto a los demás nodos. Hay que tener en cuenta que el nodo maestro estará ocupado distribuyendo el trabajo a los demás nodos, y un atraso en esta tarea, puede traer impactos negativos al cluster entero mientras los nodos esclavos pierden tiempo esperando por su siguiente instrucción.

- **Velocidad de disco**

Ya que tarde que temprano la mayor parte del trabajo del cluster será guardado como archivos en un disco duro la velocidad del disco del nodo maestro es crítica aún más si los nodos hacen uso de NFS lo que significa que cada nodo en el cluster competirá por el acceso al disco del nodo maestro. Se recomienda utilizar un tipo de tecnología SCSI sin embargo, la tecnología IDE funciona bien. Las diferencias principales son que IDE es la más usada en PCs normales, debido a que tiene un balance bastante adecuado entre precio y prestaciones. Su característica principal es que es conectada al bus de entrada/salida de la tarjeta madre de la PC, por lo que el ancho de bus es y será siempre de 16 bits. Además de ser una tecnología más económica en comparación con la SCSI. Una desventaja con respecto a SCSI, es la cantidad de dispositivos que se pueden conectar; generalmente solo 2 IDE, mientras se puede tener mas de 2 en tecnología SCSI. La tecnología SCSI ofrece, en efecto, una tasa de transferencia de datos muy alta entre el ordenador y el dispositivo SCSI. Pero aunque esto sea una cualidad muy apreciable, no es lo más importante; la principal virtud de SCSI es que dicha velocidad se mantiene casi constante en todo momento sin que el microprocesador realice apenas trabajo.

- **RAM**

La RAM es crucial en el nodo maestro por dos razones. La primera es que mientras más RAM tengamos, podemos ejecutar una mayor cantidad de procesos sin necesidad de acceder al disco. La segunda es que el kernel de Linux puede y pondrá en caché sus escrituras a disco en memoria y las mantendrá hasta que sean escritas en el disco. Ambas razones incrementan la velocidad del nodo maestro que es crucial en el desempeño total del cluster.

- **Tarjeta gráfica**

Las tarjetas gráficas o de video son los componentes encargados de crear y manejar las imágenes que vemos en nuestro monitor. Con la utilización masiva de imágenes digitales, estas tarjetas han aumentado su importancia ya que gran parte de la comodidad y de la eficacia que obtengamos en el uso de una computadora depende de ellas. En nuestro caso y debido a que se construirá un cluster para aplicaciones de realidad virtual y visualización científica este aspecto es de los más importantes a considerar.

Los aspectos a considerar para los nodos esclavos del cluster son la velocidad del procesador, RAM, velocidad de red y tarjeta gráfica dejando a un lado la capacidad de almacenamiento dado que contamos con un nodo maestro que proporciona los servicios, sistemas de archivos y aplicaciones gráficas.

3.6 Ambiente

Con clusters pequeños como el nuestro, las consideraciones del ambiente y espacio no son tan críticas con cluster de mayor tamaño digamos arriba de 128 nodos donde se consume gran cantidad de energía y debe haber un medio de disipación de calor, etc.

3.7 Arquitectura Cluster Zion

Una vez analizados todos y cada uno de los aspectos anteriores, y con base los requerimientos, presupuesto, equipo y hardware con que contamos y aclarando que para la realización del cluster el Departamento de Visualización ya había hecho un análisis previo de todos estos aspectos y adquirido el equipo

(hardware) necesario para iniciar la construcción del cluster <<dichas especificaciones se encuentran establecidas en el documento “Cluster Optimizado para Visualización Científica y Realidad Virtual (ver Anexo A)>>. La arquitectura final del cluster Zion es:

- Cluster tipo beowulf
- Cuatro nodos duales(2 procesadores Athlon MP200 a 2GHz)
- Disco duro de 100GB por nodo
- 1GB de RAM por nodo
- Tarjeta gráfica Nvidia Quadro4 900GLX por nodo
- Monitor SGI GDM-FW9011 de 21 pulgadas por nodo.
- Conectados a través de un switch 3COM con tecnología Gigabit Ethernet.
- Sistema Operativo Linux Debian 3.0 r3 “woody”

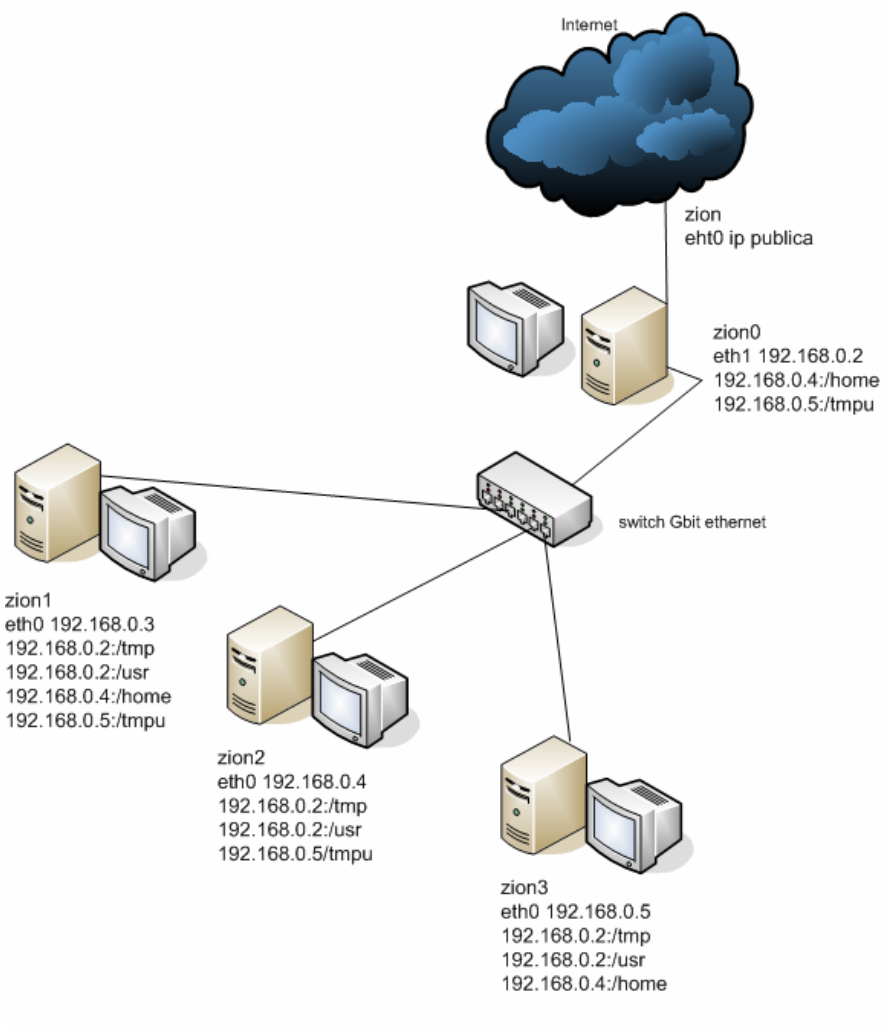

Figura 3.1 Arquitectura actual del cluster Zion

3.8 Sistema de archivos.

La definición del sistema de archivos es uno de los primeros aspectos a considerar durante la construcción de un cluster. Una buena distribución de los sistemas de archivos garantiza una mejora en rendimiento y administración del mismo. Así, y tomando en cuenta que contamos con 4 máquinas con disco duro de aproximadamente 100Gb cada una la distribución del sistema de archivos es:

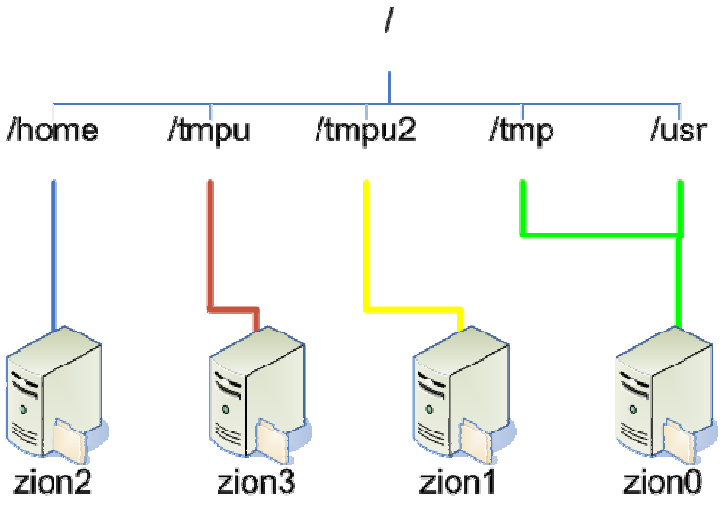

Figura 3.2. Distribución del sistema de archivos del cluster Zion.

Como puede verse en las figuras 3.1 y 3.2, cada nodo cuenta con su propia partición raíz / y cada uno de ellos comparte una partición al cluster. Se pensó con la finalidad de facilitar la administración del cluster permitiendo una utilización de espacio en disco equilibrada, aprovechar el espacio en disco de todos los nodos y para que en caso de la falla de un nodo fuera posible continuar trabajando en el cluster afectando lo menos posible la funcionalidad de éste. Así pues, se tiene un nodo como servidor de aplicaciones, otro como home y los otros como espacio para datos que se emplean en algunas visualizaciones (tmpu y tmpu2).

Zion0. Nodo maestro, será el encargado de proporcionar el sistema de archivos /usr que es el directorio principal de aplicaciones, sobre él instalaremos todas las aplicaciones solicitadas por lo que será el encargado de exportarlas al resto de los nodos. Además proporcionará el área común de /tmp.

Zion2. Nodo interno que contendrá la partición de /home y por consiguiente el directorio home de nuestros usuarios. Este nodo aunque cuenta con gran espacio en disco y podría soportar varios usuarios, se limitará empleando un esquema de cuotas de 5 Gb por usuario para no llegar al límite de su capacidad de almacenamiento.

Zion1y Zion3. Nodos internos que proporcionarán espacio en disco para ser utilizado por nuestros usuarios dado que en el nodo que proporcionará el /home estará limitado, estos dos nodos servirán como espacio temporal donde lo usuarios contarán con mayores recursos de disco para almacenar información y programas ya que algunos programas o aplicaciones gráficas requerirán gran cantidad de almacenamiento en disco para su ejecución.

La cantidad de espacio en disco o tamaño de las particiones está definido en la sección de sistema de archivos y NFS en el capítulo 5 correspondiente a la administración del cluster. A continuación se muestra el estatus actual del sistema de archivos:

```
zion0:/usr/local/sbin/admon# ./rc-ssh df -h

***** zion0 *****
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             4.6G  3.3G  1.1G  76% /
tmpfs                 505M     0  505M   0% /dev/shm
/dev/hda2              46G   33M   44G   1% /tmp
/dev/hda3              62G  4.9G   54G   9% /usr
192.168.0.4:/home      46G   29G   16G  65% /home
192.168.0.5:/tmpu     108G   93G  9.5G  91% /tmpu

***** zion1 *****
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             4.6G  848M  3.5G  20% /
/dev/hda5              98G   33M   92G   1% /tmp2
tmpfs                 504M     0  504M   0% /dev/shm
192.168.0.2:/usr       61G  4.9G   53G   9% /usr
192.168.0.4:/home      46G   29G   15G  65% /home
192.168.0.5:/tmpu     107G   93G  9.4G  91% /tmpu

***** zion2 *****
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             4.6G  1.1G  3.3G  24% /
/dev/hda2              46G   29G   15G  65% /home
tmpfs                 504M     0  504M   0% /dev/shm
192.168.0.2:/usr       61G  4.9G   53G   9% /usr
192.168.0.5:/tmpu     107G   93G  9.4G  91% /tmpu
```

```
***** zion3 *****
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             4.6G  808M  3.5G  19% /
/dev/hda3             107G   93G  9.4G  91% /tmpu
tmpfs                 504M     0  504M   0% /dev/shm
192.168.0.2:/usr       61G  4.9G   53G   9% /usr
192.168.0.4:/home      46G   29G   15G  65% /home
```

Capítulo 4. Instalación

El proceso de instalación del cluster Zion lo podemos dividir en dos etapas, la principal y más importante fue la instalación del nodo maestro y por otro lado, pero no menos importante, la correspondiente a la instalación de los nodos internos del cluster o nodos esclavos.

La instalación del nodo maestro consistió a grandes rasgos en la instalación base del sistema operativo definiendo su sistema de archivos e instalando algunas aplicaciones de desarrollo y compiladores necesarios que permitieran la creación de un kernel personalizado que se adaptara y optimizara los recursos del sistema de acuerdo al hardware, las aplicaciones y configuraciones solicitadas por el personal del Departamento de Visualización de la Dirección General de Servicios de Cómputo Académico de la UNAM; posteriormente, se continuó con la instalación de las bibliotecas de envío de mensajes PVM y MPI, algunas librerías, la instalación y configuración de la tarjeta gráfica y el ambiente gráfico del sistema para finalmente instalar las aplicaciones de realidad virtual y visualización solicitadas.

La instalación de los nodos internos consistió, al igual que en el nodo maestro, de la instalación del sistema operativo base, la instalación del kernel

optimizado generado durante la etapa de instalación del nodo maestro y la configuración necesaria de cada nodo para formar parte activa del cluster.

4.1 Instalación del nodo maestro

Antes de iniciar el proceso de instalación es necesario realizar ciertos preparativos, entre ellos, la instalación de la segunda tarjeta de red en el nodo maestro y definir el esquema de particionamiento que está directamente ligado a la estructura del sistema de archivos del cluster, más adelante en el capítulo 5 correspondiente a la Administración del cluster, se explicará a detalle el sistema de archivos, por el momento bastará con dedicarnos únicamente a la instalación base del sistema y particionamiento del nodo maestro.

La instalación del sistema operativo es una instalación típica de Debian, a continuación se hará hincapié en aquellos aspectos que se consideran de mayor importancia durante el proceso de instalación.

La versión de sistema operativo a instalar en el Cluster es Linux Debian 3.0 r3 “Woody”.

Para una mejor explicación de la instalación de Debian se recomienda consultar http://www.debian.org/releases/stable/installmanual

Entre los puntos importantes a destacar están:

- Configurar BIOS para arranque desde CD-ROM .
- Introducir el primer CD de la distribución en la unidad de CD y reiniciar el equipo.
- En la pantalla de inicio de la instalación, elegir método "bf24" para instalar con un kernel versión 2.4.18

- Elegir el idioma de instalación "en"
- Configurar el teclado.
- Crear particiones.

  / -> 4.6G

  /tmp -> 46G

  /usr -> 62G

  /swap -> 1 G
- Dar formato e iniciar partición swap. Saltarse comprobación de bloques
- Iniciar partición Linux. Elegir partición / y formato "ext3"
- ¿Montar raíz? Responder "Si"
- Instalar núcleo y módulos desde CD? Responder "Si"
- Controladores de dispositivos (módulos): sólo suele ser necesario cargar el módulo de la tarjeta de red. En nuestro caso, este módulo ya está incorporado en el kernel de instalación.
- Configuración de red. Sólo saldrá si se ha cargado correctamente el driver de la tarjeta de red.
  - Primero debe elegir el nombre del sistema o hostname. Este nombre es el que se le da a su máquina en la red. El nombre del host, para el nodo maestro será zion0.
  - Recordando que contamos con dos tarjetas de red, una 3com Gigabit Ethernet para la red interna y otra Macronix Fast Ethernet que será la interfaz al exterior, sin embargo, el kernel no reconoció la interfaz 3com por lo que sólo se podrá configurar la interfaz externa, esto nos permitirá actualizar el sistema e instalar aplicaciones desde la red, sin embargo quedará pendiente la instalación de los controladores para la interfaz interna y que permitirá la creación de la red interna del cluster.
  - IP 132.248.x.x
  - netmask 255.255.255.x

  - network 132.248.x.x
  - broadcast 132.248.x.x
  - gateway 132.248.x.x
  - El servidor de DNS será 132.248.x.x y nuestro dominio depvis.unam.mx

- Instalación del sistema base
- Hacer sistema arrancable
  - instalar LILO en el MBR (/dev/hda)
  - seleccionar poner otros S.O. en el menú de inicio de LILO
- Crear disquete de arranque: NO
- Reiniciar el sistema: extraer el CD, y arrancar con la opción Linux

Ahora comienza la parte de la configuración tras reiniciar el sistema. Esta parte puede repetirse más adelante ejecutando el comando "base-config" como administrador (root).

- Configuración hardware clock: elegir GMT->NO (sólo recomendable si también está instalado MS-Windows)
- Zona horaria: elegir "América -> México"
- Usar MD5 passwords? Elegir YES
- Usar shadow passwords? Elegir YES
- Usar solo protocolo ssh2
- Escribir clave para el administrador (root):
- Añadir un usuario normal? YES. Añadir un usuario
- Eliminar paquetes PCMCIA? Responder YES
- Usar PPP para continuar la instalación? Responder NO
- Escanear otro CD? NO
- Añadir más fuentes APT? Responder NO
- Añadir "security updates"? Responder YES

- Ejecutar “tasksel”? (Tasksel es un método de selección de paquetes a instalar) Responder YES
    - Marcar las siguientes aplicaciones:
    - NFS
    - C,C++
    - Teclear “F” (Finish).
- Ejecutar "dselect" (dselect es otro método de selcción de paquetes)? Responder NO

Ahora comenzarán a instalarse y configurarse los paquetes seleccionados. Algunos de estos paquetes preguntarán parámetros de configuración. Se sugieren las siguientes respuestas. En la mayoría de los casos valen las opciones por defecto.

- binutils: Binutils es una colección de herramientas para el desarrollo de software que contiene un ligador, un ensamblador y otras utilidades para trabajar con archivos de objeto. Responder OK
- less: paginador de archivos. Responder SI
- nfs-common: paquete que contiene las utilidades necesarias para NFS. Incluye comandos importantes como el showmount el cual permite saber los directorios exportados del Server. Responder OK
- ssh: elegir todas las opciones por defecto. Debian configurará de manera adecuada y segura nuestro servidor ssh.
- exim: servidor de correo electrónico.
    - elegir opción "5" . Que no configurará nada de exim.
- ventana de fin de instalación.

Entrar en el sistema como administrador: escribir nombre "root" en el prompt "login:"

4.2 Personalización del kernel

Como se mencionó anteriormente, las especificaciones y requerimientos del cluster fueron definidos previamente. En ellas establece el software básico así como las aplicaciones de visualización científica y realidad virtual a instalar en el cluster. Al hacer una investigación de los requerimientos y recomendaciones para instalar las diferentes aplicaciones solicitadas, encontramos que para algunas de ellas, como por ejemplo SoftGenLock, se requería de cierta configuración a nivel kernel del sistema operativo por lo que se hizo necesario la generación de un nuevo kernel (lo más ligero posible) que permitiera la ejecución de aplicaciones en tiempo real pero sobre todo que optimizará al máximo los recursos del sistema.

El núcleo o kernel es la parte básica de cualquier sistema operativo, y en él descansa el código de los servicios fundamentales para controlar el sistema entero. Básicamente, su estructura se puede clasificar en:

- Administración de procesos: Qué tareas se van a ejecutar y en qué orden y prioridad. Un aspecto importante es la planificación de CPU, cómo se optimiza el tiempo del CPU para ejecutar las tareas con el mayor rendimiento o interactividad posible con los usuarios.

- Intercomunicación de procesos y sincronización: Cómo se comunican tareas entre sí, con qué diferentes mecanismos y cómo pueden sincronizarse grupos de tareas.

- Administración entrada/salida (E/S): Control de periféricos y Administración de recursos asociados.

- Administración de memoria: Optimización del uso de la memoria, sistema de paginación y memoria virtual.

- Administración de archivos: Control y organización de los archivos presentes en el sistema y el acceso a los mismos.

Una de las particularidades de Linux es que, siguiendo la filosofía de Software Libre, nos ofrece el código fuente del propio sistema operativo (del kernel), de manera que al disponer de este código, podemos recompilarlo para adaptarlo mejor a nuestro sistema y, obtener así un mejor rendimiento.

La personalización del kernel es un proceso que requiere de conocimientos de lo que se está haciendo, además, es una de las tareas críticas, de la cual depende la estabilidad entera del sistema, por la propia naturaleza del kernel, puesto que es el elemento central del sistema. Cualquier error de procedimiento puede implicar la inestabilidad o la pérdida del sistema. Por lo tanto, no está de más llevar a cabo cualquier tarea de respaldo o backup de los datos de usuarios, datos de configuraciones que hayamos personalizado o, si disponemos de dispositivos adecuados, el backup completo del sistema.

También es recomendable disponer de algún dispositivo de arranque por si surgen problemas, o bien un disco de rescate (rescuedisk) que la mayoría de distribuciones permiten crear desde los CD de la distribución.

En algunos casos puede ser habitual la aplicación de parches (patch) al kernel. Un archivo de parche (patch file) del kernel de Linux es un archivo de texto ASCII que contiene las diferencias entre el código fuente original y el nuevo código, con información adicional de nombres de archivos y líneas de código. El programa patch sirve para aplicarlo al árbol del código fuente del kernel (normalmente en /usr/src/).

Los parches suelen necesitarse cuando un hardware o aplicación especial necesita alguna modificación en el kernel, o se han detectado algunos bugs

(errores) posteriores a alguna distribución amplia de una versión del kernel. Para corregir el problema, se suele distribuir un parche en lugar de un nuevo kernel entero.

Para el caso específico del cluster Zion, la personalización del kernel debe hacerse antes de la instalación de aplicaciones o el alta de usuarios, esto debido a que algunas aplicaciones requieren modificaciones al kernel para su óptimo desempeño.

Dentro de las aplicaciones que requieren cierta personalización del kernel se encuentra SoftGenLock que necesita de un parche (patch) al kernel llamado Realtime Application Interface (RTAI).

RTAI es una interfaz de aplicaciones en tiempo real empleada por sistemas uni-procesador y multiprocesador y permite emplear al kernel de Linux en aplicaciones de tiempo real [31].

RTAI consiste de dos partes principales: un parche al kernel de Linux que permite una capa de abstracción en el hardware y una gran cantidad de servicios que facilitan la programación en tiempo real.

SoftGenLock proporciona una solución para el genlock (sincronización del retraso del video) y el estéreo activo (a partir de la combinación de dos imágenes diferentes <una para el ojo derecho y otra para el izquierdo>, se obtenga una imagen con sensación de profundidad), sin requerir un hardware específico o modificar el código fuente del driver. SoftGenLock puede ser usado para hacer despliegue en estéreo activo en una sola o varias pantallas [32].

Dados estos requerimientos y las versiones de kernel cuyos parches son soportados, la versión del kernel a instalar en el cluster Zion será la 2.4.24.

Para la personalización y construcción de nuestro nuevo kernel, es necesario descargar los paquetes ncurses, y kernel-package. El paquete ncurses son librerías que ayudan a la programación en modo texto para Linux, se requiere para realizar la configuración y prender o apagar módulos en nuestro kernel. El paquete kernel-package permite la creación de un paquete .deb del kernel a compilar.

```
apt-get install ncurses* kernel-package
```

Ir al directorio /usr/src

```
cd /usr/src
```

Descargar los archivos fuente del kernel

```
wget http://www.kernel.org/pub/linux/kernel/v2.4/linux-2.4.24.tar.bz2
```

Desempaquetar el kernel

```
tar xvfzj linux-2.4.24.tar.bz2
```

Crear la liga:

```
ln -s linux-2.4.-24 linux
cd linux
```

Descargar y aplicar el parche para RTAI

```
patch -p1 /temp/rtai-3.0r3/rtai-core/arch/i386/patches/legacy-2.4.24.patch
```

- Configuración del kernel

```
make menuconfig
```

- Habilitar los siguientes módulos:

```
rtai
iptables    ipt_state    iptable_filter    ipt_multiport    ipt_MASQUERADE
iptables_nat ipt_REJECT iptable_mangle ip_log
nfs
tulip (para la tarjeta de red)
ext3
smp
devfs
quota
```

- Una vez habilitados todos los módulos requeridos, crear la imágen del kernel en paquete debian. Al ejecutar el siguiente comando se inicia la compilación del kernel y finalmente se crea una imagen de kernel en un paquete .deb que debe ser instalado empleando en comando dpkg -i.

```
fakeroot make-kpkg --revision=custom.rthal5_10.00 kernel_image
```

- Esto generará una imagen del kernel en el directorio */usr/src* con el nombre: `kernel-image-2.4.24-rthal5_10.00.Custom_i386.deb`. Esta será la imagen del kernel optimizado que deberá ser instalada en todos y cada uno de los nodos del cluster.

- Instalar la imagen de kernel generada

```
dpkg -i ../kernel-image-2.4.24-rthal5_10.00.Custom_i386.deb
```

- Se preguntará si se quiere modificar el archivo lilo.conf contestar que sí. Esto generará una entrada en el gestor de arranque para poder iniciar con la nueva imagen del kernel.

- Reiniciar el sistema

```
sync;sync;sync; shutdown -r now
```

- Una vez que el sistema haya reiniciado correctamente, se deben instalar y configurar los controladores de la tarjeta gráfica y de la segunda tarjeta de red 3com.

  El driver para la tarjeta de red es: bcm5700

  Descargar los fuentes, compilar e instalar.

```
cd bcm
make
make install
```

4.3 Tarjeta gráfica

Los controladores para la tarjeta Nvidia Quadro4 900GLX son proporcionados por el fabricante. El controlador instalado es la versión 1.0-5336. El 15 de febrero de 2005 se actualizó a la versión 1.0-6629. Los pasos para instalarlos son:

- Descargar el archivo
  - `wget http://download.nvidia.com/XFree86/Linux-x86/1.0-5336-pkg1.run`
- Ejecutar el script.
  - `sh NVIDIA-Linux-x86-1.0-5336-pkg1.run`
- Para el uso del controlador, es necesario modificar algunas líneas del archivo de configuración /etc/X11/XF86Config-4
  - Sustituir la línea
    - `Driver "nv" (or Drive "vesa") por`
    - `Driver "nvidia"`
  - Asegurarse que en la sección Module se tenga la línea
    - `Load "glx"`
  - Remover en las sección modules las líneas:
    - `Load "dri"`
    - `Load "GLcore"`
- El archivo se encuentra en el Anexo B.

4.4 Instalación de las Bibliotecas de envío de mensajes MPICH y LAM

El proceso de instalación de estas librerías fué bastante sencillo ya que dichas librerías se encuentran ya dentro de los paquetes Debian disponibles, así la instalación consistió en:

Instalación de mpi, lam, xmpi y pvm desde paquetes deb empleando apt-get install.

```
zion0:~# apt-get install lam4
zion0:~# apt-get install mpich
zion0:~# apt-get install pvm
```

Configurar lam

- Agregar en /etc/lam/lam-bhost.def las ip de los nodos (incluyendo al maestro) este archivo debe estar en todos los nodos.

```
zion0:/etc/lam# cat lam-bhost.def
# Copyright (c) 2001-2003 The Trustees of Indiana University.
#                         All rights reserved.
# Copyright (c) 1998-2001 University of Notre Dame.
#                         All rights reserved.
# Copyright (c) 1994-1998 The Ohio State University.
#                         All rights reserved.
#
# This file is part of the LAM/MPI software package.  For license
# information, see the LICENSE file in the top level directory of the
# LAM/MPI source distribution.
#
# $HEADER$
#
#       Function:       - bhost.def
#                       - a single-node LAM
#

192.168.0.2
192.168.0.3
192.168.0.4
192.168.0.5
```

- Ejecutar para comprobar que esté funcionando una vez que se encuentren instalados y configurados todos los nodos del cluster.
    - **recon**. Comando para comprobar si el cluster está preparado para ser arrancado. Utiliza las variables de entorno LAMHOME para determinar donde se encuentra la instalación de LAM y LAMRSH para indicar que la conexión a los nodos se realize usando SSH.

```
cesar@zion0:~$ recon
-----------------------------------------------------------------------
----
Woo hoo!

recon has completed successfully.  This means that you will most likely
be able to boot LAM successfully with the "lamboot" command (but this
is not a guarantee).  See the lamboot(1) manual page for more
information on the lamboot command.

If you have problems booting LAM (with lamboot) even though recon
worked successfully, enable the "-d" option to lamboot to examine each
step of lamboot and see what fails.  Most situations where recon
succeeds and lamboot fails have to do with the hboot(1) command (that
lamboot invokes on each host in the hostfile).
-----------------------------------------------------------------------
----
```

    - **lamboot**. Comando usado para arrancar el cluster. La configuración del cluster debe la obtiene del fichero $HOME/etc/lam-bhost.def. Utiliza las variables de entorno LAMHOME para determinar donde se encuentra la instalación de LAM y LAMRSH para indicar que la conexión a los nodos se realize usando SSH.

```
cesar@zion0:~$ lamboot

LAM 7.1.1/MPI 2 C++/ROMIO - Indiana University
```

    - **lamnodes**. Lista el conjunto de nodos que forman el cluster

```
cesar@zion0:~$ lamnodes
n0     zion0:1:origin,this_node
n1     zion1:1:
n2     zion2:1:
n3     zion3:1:
```

4.5 Instalación de los nodos internos

Una vez que el nodo maestro fue instalado debemos instalar el resto de los nodos que conformarán nuestro cluster. En este sentido existen una gran cantidad de métodos que optimizan el proceso de instalación como son la instalación desde red, FAI (Fully Automatic Installation. Herramiena de instalación automática para Debian), etc. Sin embargo, dadas las características tan específicas con que va a contar el cluster Zion, se decidió instalar uno a uno cada nodo del cluster.

Un aspecto importante a consideración en la instalación de los nodos internos es que su esquema de particionamiento es diferente con el nodo maestro y entre ellos, la forma de particionar el nodo maestro y los nodos internos se encuentra definida en la sección Sistema de Archivos del capítulo Consideraciones de diseño.

Así pues la instalación de los nodos internos consistió en:

- La instalación de un sistema operativo base que sólo contara con una partición / y la partición correspondiente al nodo definido en el esquema del sistema de archivos.
- La instalación del kernel personalizado generado para el nodo maestro.
- La instalación de los drivers de red y tarjeta gráfica
- Todos estos pasos han sido documentados a detalles en el apartado anterior.
- Dentro del capítulo de administración se muestra la configuración necesaria de los nodos hijos para ser parte activa del cluster.

Capítulo 5. Administración

Una vez que el Cluster ha sido instalado y configurado, debe garantizarse su funcionamiento de manera eficiente, para ello, deben llevarse a cabo varias tareas que en conjunto reciben el nombre de administración del sistema. Podríamos separar las tareas de administración en dos partes principales: administración local del sistema y administración de red [1].

5.1 Administración Local

Dentro de las tareas de administración local del sistema tenemos:

5.1.1 Arranque y apagado del sistema

Cualquier sistema basado en UNIX tiene sistemas de arranque y apagado valorables, de manera que podemos configurar qué servicios ofrecemos en el arranque de la máquina y cuándo hay que pararlos, o programar el apagado del sistema para su mantenimiento.

5.1.2 Administración de usuarios y grupos.

Dar cabida a los usuarios es una de las principales tareas de cualquier administrador. Habrá que decidir qué usuarios podrán acceder al sistema, de

qué forma, bajo qué permisos y establecer los grupos de usuarios a crear y administrar. El objetivo de los usuarios, grupos y permisos es brindar privacidad y organización permitiendo también compartir información cuando así se desea.

En el caso del cluster Zion, se han creado tres grupos principales de usuarios: RV, LabVis y admin. Dentro de los usuarios en el grupo RV se encontrarán aquellos que requieran acceso a aplicaciones de realidad virtual, en LabVis los de visualización científica y en admin los administradores del sistema permitiendo de esta manera asignar perfiles de usuarios dependiendo de las aplicaciones a utilizar; así, usuarios del grupo RV tendrán variables de ambiente asignadas diferentes a las de usuarios que pertenezcan al grupo LabVis o admin.

5.1.2.1 Alta de usuarios

Uno de los grandes problemas con los que nos enfrentamos al momento de administrar usuarios, es que debido a las características de nuestro cluster, cada nodo contiene un archivo /etc/passwd independiente del resto sin embargo, no podemos perder de vista que contamos con un nodo que nos provee de la partición /home. Tomando esto en consideración, y con el fin de facilitar la administración de usuarios, se decidió tomar al nodo 2 (zion2) como el que será el encargado de replicar los archivos de usuarios y grupos y manipular todo lo referente a la administración de los usuarios. Para tal efecto, se elaboró un script en perl de alta de usuarios; entre otras funcionalidades y como acciones previas, el script debe verificar primero que todos y cada uno de los nodos del cluster se encuentren funcionando (prendidos), además, como una de las restricciones anteriormente citadas, para poder dar de alta un usuario se debe estar logeado en el nodo 2 (zion2) y con privilegios de root. Así, una vez verificadas cada una de estas condiciones, se solicita primero el grupo al que pertenecerá el usuario y posteriormente se emplea el comando adduser para añadirlo al sistema. Hasta

este momento, el usuario se encuentra dado de alta sólo en el nodo que contiene el /home por lo que aún falta hacer la réplica de los archivos relacionados como son /etc/passwd, /etc/group y /etc/shadow al resto de los nodos empleando para ello rsync. El script se llama alta_usuario y se encuentra en el Anexo B.

Como parte de la administración de los usuarios y como actividad pendiente está la de realizar un script que automatice la generación de llaves ssh una vez que ha sido dado de alta un usuario. Esto para permitir la autenticación directa en cada uno de los nodos sin requerir de password para ello. Actualmente la generación de dichas llaves se realiza y replica manualmente a los nodos.

5.1.2.2 Baja de usuarios

Dentro del proceso de baja de usuarios se pensó en dos aspectos principales, la inhabilitación de la cuenta de usuario empleando para ello el comando usermod con la opción –L (lock) o bien el borrado del usuario y todos y cada uno de sus archivos que tiene dentro del cluster. El script se llama baja_usuario y se encuentra en el Anexo B.

5.1.3 Administración de recursos del sistema

Este punto se refiere a cuentas, permisos de usuarios, aplicaciones instaladas y servicios, en resumen qué ofrecemos, cómo lo ofrecemos y a quién damos acceso.

Uno de los aspectos a considerar en este punto es que debido a las múltiples aplicaciones instaladas a utilizar, la gran mayoría de ellas requiere de variables de ambiente que permiten trabajar con ellas; para evitar que los usuarios tengan que hacer manualmente la exportación de dichas variables, se agregan en el archivo ~/.bashrc así, el archivo ~/.bashrc deberá contener las siguientes líneas:

```
#-----------------------------
#-----------------------------
# Variables definidas
#-----------------------------
#-----------------------------

# lam/mpi
export LAMBHOST=/etc/lam/lam-bhost.def

# vrjuggler 2.0-alpha3
#export VJ_BASE_DIR=/home/marce/temporal/vrjuggler-2.0-alpha3.src/instlinks
#export VJ_DEPS_DIR=/usr/local
export JDK_HOME=/usr/local/j2sdk1.4.2_04
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$VJ_BASE_DIR/lib

# vrjuggler 2.0-alpha4
#export VJ_BASE_DIR=/usr/local/vrjuggler-2.0-alpha4.src
export VJ_DEPS_DIR=/usr/local

# vrjuggler 1.0- 7
export VJ_BASE_DIR=/usr/local/vrjuggler

# vrjuggler para netjuggler
export VJ_BASE_DIR=/usr/local/Net-vrjuggler
export NJ_BASE_DIR=/usr/local/Netjuggler
export USE_NETJUGGLER=yes
export PATH=$PATH:$VJ_BASE_DIR/bin:$NJ_BASE_DIR/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/X11R6/:$VJ_BASE_DIR/lib

# OpenSceneGraph
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
export   OSG_FILE_PATH=/usr/local/OpenSceneGraph-Data:/usr/local/OpenSceneGraph-
Data/Images:/usr/local/OpenSceneGraph-Data/fonts

# Syzygy
export SZGHOME=/usr/local/szg-0.61
export SZGEXPT=/usr/local/szgexpt-0.61
export SZGDATA=/usr/local/szg-0.61/Data
```

```
#vrNav2-5.8
export VRNAV_HOME=/usr/local/vrNav2-5.8

# Dx
export DXDATA=/usr/local/dx-samples/dx/samples/data
export DXROOT=/usr/local/dx-4.3.2/dx
export PATH=$PATH:/usr/local/dx-4.3.2/dx/bin_linux/:/usr/local/dx-4.3.2/bin

# VTK
export VTK_DATA=/usr/local/vtk/VTKData-release-4-2/Data
export VTK_DIR=/usr/local/vtk/include

# Vis5d
export PATH=$PATH:/usr/local/vis5d-5.1
```

Por otra parte y dado que una de las funcionalidades principales del cluster es poder realizar sesiones gráficas simultáneas en cada nodo y poder hacer el despliegue de una imágen en cada uno de ellos, es necesario que los usuarios puedan ejecutar sesiones gráficas, esto representó algunos problemas al principio, sin embargo se han solucionado agregando las siguientes lineas al archivo .bash_profile de cada usuario.

```
if [ -f ~/.bashrc ]; then
   # leer Variables
    source ~/.bashrc
fi

if [ "$TERM" = "linux" -a `tty` = "/dev/tty1" ]; then
    HOST=`hostname`;
    export XAUTHORITY=$HOME/.Xauthority-$HOST
    if [ "$HOST" = "zion1" ]; then
        /usr/bin/X11/startx /usr/bin/blackbox -- :1;
    elif [ "$HOST" = "zion2" ]; then
        /usr/bin/X11/startx /usr/bin/blackbox -- :2;
    elif [ "$HOST" = "zion3" ]; then
        /usr/bin/X11/startx /usr/bin/blackbox -- :3;
    fi

fi
```

En términos generales lo que hacen estas líneas es iniciar la sesión gráfica utilizando el manejador de ventanas blackbox en cada uno de los nodos del cluster para así contar con un ambiente gráfico simultáneo en cada nodo del cluster y poder ver el despliegue gráfico de una aplicación en las 4 pantallas con las que contamos.

Una herramienta más desarrollada para la administración es la actualización o sincronización de ciertos archivos de configuración en cada nodo del cluster

Esta tarea generalmente es realizada por NIS (Network Information Service) sin embargo, debido a cuestiones de seguridad (passwords en claro), NIS no ha sido implementado en el cluster por lo que se requiere de un mecanismo paralelo que realice la función de sincronización de archivos en los nodos del cluster.

El método implementado para esta tarea es un shell script llamado ssync que sincronizará todos los archivos listados en el archivo /etc/sfiles en cada uno de los nodos listados en el archivo /etc/nodos. Así pues el código de este shell script es:

```
zion0:~# cat /usr/local/sbin/admon/ssync
#!/bin/bash
#
# Este script sinconiza archivos en  los nodos
# "escalvos" del cluster. Lee el archivo /etc/nodos que contiene
# los nodos activos del cluster y copia los archivos
# definidos en /etc/sfiles
#
for nodo in $(cat /etc/nodos);
do
   echo "Sincronizando $nodo"
   for file in $(cat /etc/sfiles);
   do
      echo "Copiando  $file"
      rsync $file $nodo:$file
   done
```

```
done
```

El contenido de los archivos /etc/sfiles y /etc/nodos es:

```
zion0:~# cat /etc/nodos
zion0
zion1
zion2
zion3
```

Para el archivo /etc/sfiles

```
zion0:~# cat /etc/sfiles
/etc/passwd
/etc/shadow
/etc/group
/etc/skel/.alias
/etc/skel/.bash_profile
/etc/skel/.bashrc
/etc/skel/.cshrc
/etc/shutdown.allow
```

El script ssync puede ser ejecutado cada vez que sea necesario o bien incluirlo como un trabajo calendarizado tipo cron que se esté ejecutando cada 5 minutos.

```
zion0:~# crontab -l
# Sincronizar cada 5 minutos los archivos a c/u de los nodos
05 00 * * * /usr/local/sbin/admon/ssync
```

De esta manera, se pueden añadir o quitar archivos a sincronizar y ejecutando el script en un sólo nodo (para nuestro caso el nodo maestro zion0) garantizamos que la información almacenada en archivos de configuración de nuestro interés sea la misma en cada uno de los nodos del cluster.

Por otra parte, durante la administración del cluster o incluso los usuarios mismos necesitan ejecutar comandos en cada uno de los nodos del cluster para verificar procesos, eliminar trabajos, verificar la carga del sistema, apagar el cluster, etc. La forma de realizar esta tarea es mediante un shell script llamado

rc-ssh que al igual que el script ssync utiliza el archivo /etc/nodos para ejecutar el comando en los nodos listados en dicho archivo. El script es el siguiente:

```
zion0:~# cat /usr/local/sbin/admon/rc-ssh
#!/bin/bash
# Script para ejecutar en c/u de los nodos definidos
# en el archivo /etc/nodos
# un comando determinado
# Sintaxis:
# /usr/local/sbin/admon/rc-ssh comando
#
for nodo in $(cat /etc/nodos) ;
do
   echo
   echo "***** $nodo *****"
   /usr/bin/ssh $nodo $*
done
```

Como puede verse del código anterior, la forma en que se ejecutan los comandos en cada uno de los nodos es mediante la ejecución de comandos remotos utilizando secure shell (ssh), aquí se presenta un problema, únicamente con comandos administrativos que requieren privilegios de super-usuario para su ejecución ya que como medida de seguridad, el cluster no permiten el acceso al nodo maestro como el super-usuario root. La solución a este problema consistió en la creación de un segundo demonio de ssh que escuchara por la interfaz interna en el nodo maestro. Así, se hizo una copia del archivo /etc/init.d/ssh a /etc/init.d/ssh-1 modificando en éste último la línea:

```
start-stop-daemon --start --quiet --pidfile /var/run/sshd.pid --exec
/usr/sbin/sshd -- $SSHD_OPTS
```

por :

```
start-stop-daemon --start --quiet --pidfile /var/run/sshd-1.pid --exec
/usr/sbin/sshd -f /etc/ssh/sshd_config-1
```

El archivo /etc/ssh/sshd_config-1 es una copia del archivo /etc/ssh/sshd_config modificando en el primero las líneas:

```
ListenAddress 132.248.X.X
```

por:

```
ListenAddress 192.168.0.2
```

y

```
PermitRootLogin no
```

por:

```
PermitRootLogin yes
```

Con estas modificaciones se agrega a los scripts de inicialización del sistema el demonio ssh-1 garantizando que en la interfaz externa del cluster se encuentre restringido el acceso como root pero permitiéndolo para la red interna logrando así ejecutar comandos que requieren privilegios administrativos empleando el script rc-ssh.

Así, un ejemplo de la ejecución de este comando:

```
zion0:/usr/local/sbin/admon# ./rc-ssh df -h
***** zion0 *****
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             4.6G  3.3G  1.1G  76% /
tmpfs                 505M     0  505M   0% /dev/shm
/dev/hda2              46G   33M   44G   1% /tmp
/dev/hda3              62G  4.9G   54G   9% /usr
192.168.0.4:/home      46G   29G   16G  65% /home
192.168.0.5:/tmpu     108G   93G  9.5G  91% /tmpu

***** zion1 *****
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             4.6G  848M  3.5G  20% /
/dev/hda5              98G   33M   92G   1% /tmp2
tmpfs                 504M     0  504M   0% /dev/shm
192.168.0.2:/usr       61G  4.9G   53G   9% /usr
192.168.0.4:/home      46G   29G   15G  65% /home
192.168.0.5:/tmpu     107G   93G  9.4G  91% /tmpu

***** zion2 *****
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             4.6G  1.1G  3.3G  24% /
/dev/hda2              46G   29G   15G  65% /home
tmpfs                 504M     0  504M   0% /dev/shm
192.168.0.2:/usr       61G  4.9G   53G   9% /usr
192.168.0.5:/tmpu     107G   93G  9.4G  91% /tmpu
```

```
***** zion3 *****
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1             4.6G  808M  3.5G  19% /
/dev/hda3             107G   93G  9.4G  91% /tmpu
tmpfs                 504M     0  504M   0% /dev/shm
192.168.0.2:/usr       61G  4.9G   53G   9% /usr
192.168.0.4:/home      46G   29G   15G  65% /home
```

5.1.4 Administración del sistema de archivos

El cluster como tal dispone de diferentes recursos de almacenamiento de datos con diferentes sistemas de acceso a los archivos. Pueden ser permanentes o extraíbles o temporales, con lo cual habrá que modelar y gestionar los procesos de montaje y desmontaje de los sistemas de archivos que ofrezcan los discos o dispositivos afines. Este punto será tocado más adelante dentro de la sección de Network File System (NFS).

5.1.5 Cuotas del sistema

Cualquier recurso que vaya a ser compartido tiene que ser administrado, y según la cantidad de usuarios, habrá que establecer un sistema de cuotas para evitar el abuso de los recursos por parte de los usuarios o establecer clases (o grupos) de usuarios diferenciados por mayor o menor uso de recursos. Suelen ser habituales sistemas de cuotas de espacio de disco, o de impresión, o de uso de CPU (tiempo de cómputo usado). Para el caso del cluster Zion, sólo se cuenta con sistema de cuotas de espacio en disco.

El sistema de cuotas sólo aplica al espacio que nuestros usuarios van a tener para almacenar su información por lo que la configuración de cuotas en disco sólo se llevará a cabo en el nodo 2 que proporciona el directorio home de los usuarios. Se asignará un total de 3Gb a cada usuario en su partición /home, en caso de requerir más espacio se cuenta con las particiones /tmpu y /tmpu2.

Lo primero a hacer es habilitar el soporte para cuotas en el kernel, proceso que se realizó durante la personalización del kernel. EL siguiente paso es editar el archivo /etc/fstab del nodo 2. Se agrega usrquota y grpquota a la partición home.

```
zion2:~# cat /etc/fstab
# /etc/fstab: InformaciÃ³n estÃ¡tica del sistema de archivos.
#
# <Sis. archivos>       <Punto montaje> <Tipo>  <Opciones>
<volcado>       <pasada>
/dev/hda1       /               ext3    errors=remount-ro       0       1
/dev/hda2       /home           ext3    errors=remount-ro,usrquota,grpquota
0       1
/dev/hda4       none            swap    sw                      0       0
proc            /proc           proc    defaults                0       0
/dev/fd0        /floppy         auto    user,noauto             0       0
/dev/cdrom      /cdrom          iso9660 ro,user,noauto          0       0

192.168.0.2:/tmp        /tmp    nfs     rw,hard,intr            0       0
192.168.0.2:/usr        /usr    nfs     rw,hard,intr            0       0
192.168.0.5:/tmpu      /tmpu   nfs     rw,hard,intr            0       0
```

Ahora utilizaremos el comando `quotacheck`, el cual examina los sistemas de archivos con cuotas activadas y construye una tabla del uso del disco por sistema de archivo. La tabla es luego usada para actualizar la copia del uso del disco del sistema operativo. Además, los archivos de cuotas de disco del sistema de archivos, son actualizados. Esto genera dos archivos en /home con el nombre `/home/aquota.user` y `/home/aquota.group`. Cambiar los permisos al archivo aquota para que sólo pueda ser editado por el superusuario y evitar modificaciones en los límites de cuotas por parte de los usuarios. `chmod 600 /home/aquota.*`. Finalmente se deben asignar el espacio de cuota en disco esto se logra con el comando `edquota –u usuario` o bien `edquota –g grupo`.

Por ejemplo:

```
zion2:~# edquota -u cesar
  Disk quotas for user cesar (uid 1001):
```

```
Filesystem      blocks     soft     hard     inodes     soft     hard
/dev/hda2       2306072  2900000  3000000      29677        0        0
```

5.1.6 Backup y restauración del sistema

Backup y restauración del sistema: es necesario establecer políticas periódicas (según importancia de los datos), de copias de seguridad de los sistemas. Hay que establecer periodos de copia que permitan salvaguardar nuestros datos, de fallos del sistema (o factores externos) que puedan provocar pérdidas o corrupción de datos.

El esquema de respaldos del cluster consiste en la generación de un respaldo del nodo maestro cada vez que se instala una nueva aplicación, o bien mensualmente. El respaldo se realiza generando un archivo .tgz y quemándolo en un cd. El respaldo del espacio de /home de los usuarios queda a responsabilidad de estos dado que no se cuenta con mecanismos para poder respaldar tanta información.

5.1.7 Automatización de tareas rutinarias

Muchas de las tareas rutinarias de la administración o del uso habitual de la máquina pueden ser fácilmente automatizables, ya sea debido a su simplicidad (y por lo tanto, a la facilidad de repetirlas), como a su temporización, que hace que tengan que ser repetidas en periodos concretos. Estas automatizaciones suelen hacerse, bien mediante programación por lenguajes interpretados de tipo script (shells, Perl, etc.), como por la inclusión en sistemas de temporización (crontab, y at). Por el momento, sólo se cuenta con la sincronización de archivos empleando el script rc-ssh mencionado con anterioridad.

5.1.8 Seguridad del sistema

Podemos dividir la seguridad en dos grandes ramas la seguridad local y la seguridad en red, ambas tratan sobre protecciones a los recursos frente a usos indebidos o accesos no permitidos a datos del sistema o de otros usuarios o grupos ya sea local o remotamente. Este tema es tratado a profundidad en el Capítulo 6.

5.1.9 Monitoreo del sistema

Para poder verificar el funcionamiento correcto de nuestro sistema, es necesario llevar políticas de análisis de bitácoras (logs) que nos puedan informar de los posibles fallos del sistema o del rendimiento que se obtiene de una aplicación, servicio o recurso hardware. O bien, resumir los recursos gastados, los usos realizados o la productividad del sistema en forma de informe. Para llevar a cabo esta tarea, se instaló la herramienta llamada ganglia en el cluster, la instalación, configuración y análisis del monitoreo del sistema, se explica en el capítulo 7.

5.1.10 Ajuste y personalización

Con esto nos referimos a las técnicas de optimización del sistema para un fin dado. Suele ser habitual que un sistema esté pensado para una tarea concreta y que podamos verificar su funcionamiento adecuado (por ejemplo, mediante logs), para examinar sus parámetros y adecuarlos a las prestaciones que se esperan. Con personalización nos referimos a la reconfiguración del kernel del sistema operativo; los kernels son altamente personalizables, según las características que queramos incluir y el tipo de dispositivos que tengamos o esperemos tener en nuestra máquina, así como los parámetros que afecten al

rendimiento del sistema o que consigan las aplicaciones. Este paso ha sido explicado a detalle dentro en el capítulo anterior.

5.2 Administración de red

En término generales, la administración de red comprende el tipo de interfaz de red que utilizamos, ya sea el acceso a una red local, la conexión a una red mayor, o conexiones del tipo banda ancha con tecnologías DSL o RDSI. Además, el tipo de conectividades que vamos a tener, en forma de servicios o peticiones, los datos que circularán, de dónde o hacia dónde se dirigirán, dependiendo de los dispositivos de red disponibles y de las funciones de la máquina en red; posiblemente, será necesario redirigir el tráfico desde o hacia uno o más sitios. También es necesario considerar la seguridad de red, una red, sobre todo si es abierta (como Internet) a cualquier punto exterior, es una posible fuente de ataques y, por lo tanto, puede comprometer la seguridad de nuestros sistemas o los datos de nuestros usuarios. Hay que protegerse, detectar e impedir posibles ataques con una política de seguridad clara y eficaz.

La arquitectura de red a seguir en el cluster es la siguiente:

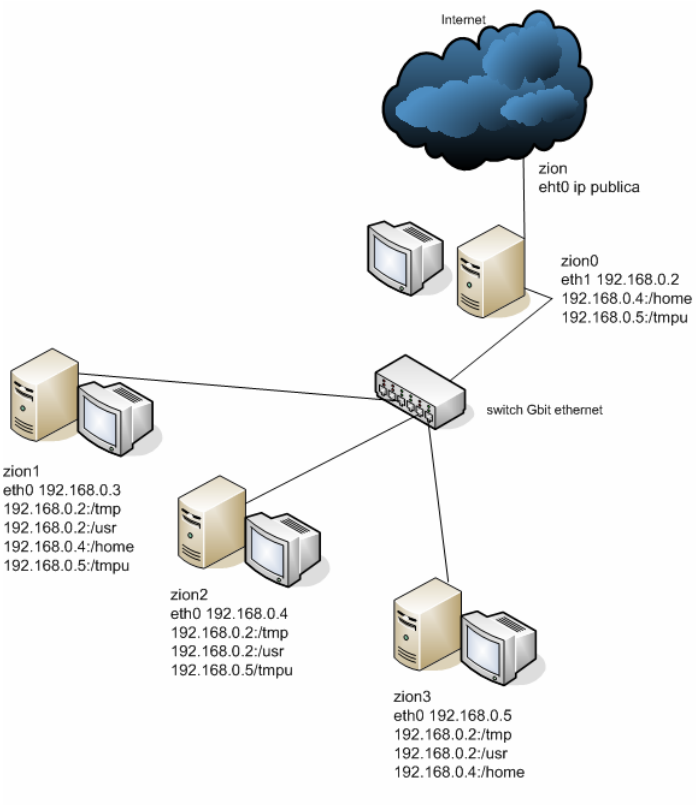

Figura 5.1 Arquitectura de red del cluster Zion

Como puede verse en la figura anterior, la red del cluster es una red interna Gigabit ethernet; el nodo maestro cuenta con dos interfaces de red una de ellas conectada a la red interna y la otra con una ip pública que permitirá el acceso al cluster desde el exterior. Esto implica emplear al nodo maestro como un tipo de proxy/firewall que permita a los nodos internos salir al exterior con el propósito de poder hacer respaldos remotos de los nodos. La configuración de esta topología, se llevó a cabo dentro de un script de arranque que realiza la función de proxy/nat y a su vez de firewall que protege nuestra red interna.

La interfaz de red que será la puerta al exterior será eth0 (100Mb/s), mientras que la interna que se comunicará con los demás nodos del cluster será eth1 (3com a 1Gb/s). La configuración del nodo maestro como Proxy para permitir salida al exterior a los nodos internos está implementada en el script del

firewall personal en el siguiente capítulo. El archivo para la configuración de las tarjetas de red es el siguiente:

```
zion0:/usr/local/sbin/admon# ./rc-ssh cat /etc/network/interfaces
***** zion0 *****
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

# The first network card - this entry was created during the Debian
installation
# (network, broadcast and gateway are optional)
# Macronix MAC - 0080AD888778
auto eth0
iface eth0 inet static
        address 132.248.x.x
        netmask 255.255.255.224
        network 132.248.x.0
        broadcast 132.248.x.255
        gateway 132.248.x.x

# 3com 3C996B MAC - 000476F6a509
auto eth1
iface eth1 inet static
        address 192.168.0.2
        netmask 255.255.255.0
        network 192.168.0.0
        broadcast 192.168.0.255
        gateway 132.248.x.x

***** zion1 *****
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
        address 192.168.0.3
        netmask 255.255.255.0
        network 192.168.0.0
        broadcast 192.168.0.255
        gateway 192.168.0.2
```

```
***** zion2 *****
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
        address  192.168.0.4
        netmask 255.255.255.0
        network 192.168.0.0
        broadcast 192.168.0.255
        gateway 192.168.0.2

***** zion3 *****
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
        address 192.168.0.5
        netmask 255.255.255.224
        network 192.168.0.0
        broadcast 192.168.0.255
        gateway 192.168.0.2
```

Asignar servidor dns. Crear el archivo /etc/resolv.conf y añadir las líneas:

```
search depvis.unam.mx
nameserver 132.248.x.x
```

NFS (Network Fylesystem).

A menudo, en las estructuras de sistemas en red es necesario compartir información (como los propios archivos) por parte de todos o algunos de los usuarios. O sencillamente, debido a la distribución física de los usuarios, es necesario un acceso a los archivos desde cualquier punto de la red. Los

sistemas de archivos por red (como NFS) permiten un acceso transparente a los archivos, independientemente de nuestra situación en la red.

El sistema de archivos está basado en la siguiente figura:

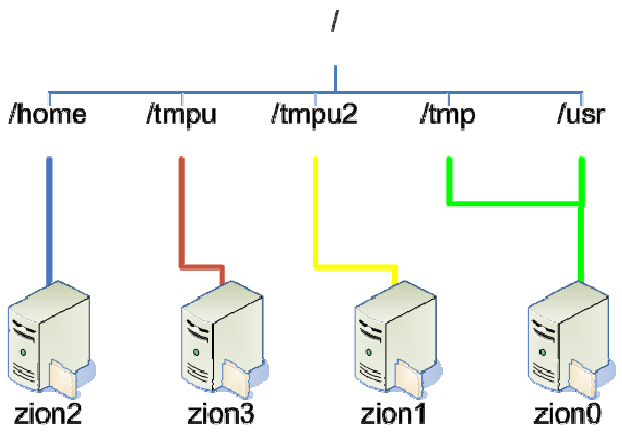

Figura 5.2. Distribución del sistema de archivos del cluster Zion.

Para llevar a cabo esto se debe modificar el archivo /etc/fstab con las siguientes líneas en el nodo maestro y nodos internos.

```
zion0:/usr/local/sbin/admon# ./rc-ssh cat /etc/fstab

***** zion0 *****
# /etc/fstab: Información estática del sistema de archivos.
#
# <Sis. archivos>  <Punto montaje>  <Tipo>  <Opciones>  <volcado>  <pasada>
/dev/hda1        /                ext3     errors=remount-ro        0        1
/dev/hda5        none             swap     sw                       0        0
proc             /proc            proc     defaults                 0        0
/dev/fd0         /floppy          auto     user,noauto              0        0
/dev/cdrom       /cdrom           iso9660  ro,user,noauto           0        0
/dev/hda2        /tmp     ext3    defaults                          0        2
/dev/hda3        /usr     ext3    defaults                          0        2

192.168.0.4:/home        /home    nfs      rw,hard,intr             0        0
192.168.0.5:/tmpu        /tmpu    nfs      rw,hard,intr             0        0


***** zion1 *****
# /etc/fstab: InformaciÃ³n estÃ¡tica del sistema de archivos.
```

```
#
# <Sis. archivos>       <Punto montaje> <Tipo>  <Opciones>
<volcado>       <pasada>
/dev/hda1       /               ext3    errors=remount-ro       0       1
/dev/hda2       none            swap    sw                      0       0
proc            /proc           proc    defaults                0       0
/dev/fd0        /floppy         auto    user,noauto             0       0
/dev/cdrom      /cdrom          iso9660 ro,user,noauto          0       0
/dev/hda5       /tmp2           ext3    defaults                0       0

192.168.0.2:/tmp       /tmp    nfs     rw,hard,intr            0       0
192.168.0.2:/usr        /usr    nfs     rw,hard,intr            0       0
192.168.0.4:/home       /home   nfs     rw,hard,intr            0       0
192.168.0.5:/tmpu      /tmpu   nfs     rw,hard,intr            0       0



***** zion2 *****
# /etc/fstab: InformaciÃ³n estÃ¡tica del sistema de archivos.
#
# <Sis. archivos>       <Punto montaje> <Tipo>  <Opciones>
<volcado>       <pasada>
/dev/hda1       /               ext3    errors=remount-ro       0       1
/dev/hda2       /home           ext3    errors=remount-ro,usrquota,grpquota
0       1
/dev/hda4       none            swap    sw                      0       0
proc            /proc           proc    defaults                0       0
/dev/fd0        /floppy         auto    user,noauto             0       0
/dev/cdrom      /cdrom          iso9660 ro,user,noauto          0       0

192.168.0.2:/tmp        /tmp    nfs     rw,hard,intr            0       0
192.168.0.2:/usr        /usr    nfs     rw,hard,intr            0       0
192.168.0.5:/tmpu      /tmpu   nfs     rw,hard,intr            0       0


***** zion3 *****
# /etc/fstab: InformaciÃ³n estÃ¡tica del sistema de archivos.
#
# <Sis. archivos>       <Punto montaje> <Tipo>  <Opciones>
<volcado>       <pasada>
/dev/hda1       /               ext3    errors=remount-ro       0       1
/dev/hda2       none            swap    sw                      0       0
proc            /proc           proc    defaults                0       0
/dev/fd0        /floppy         auto    user,noauto             0       0
/dev/cdrom      /cdrom          iso9660 ro,user,noauto          0       0
/dev/hda3       /tmpu   ext3    defaults                        0       2

192.168.0.2:/tmp       /tmp    nfs     rw,hard,intr            0       0
```

```
192.168.0.2:/usr        /usr    nfs     rw,hard,intr            0       0
192.168.0.4:/home       /home   nfs     rw,hard,intr            0       0
```

Ahora dado que también nuestros nodos exportarán sistemas de archivos, se requiere modifcar el archivo /etc/exports. Es decir, para que el resto de los nodos puedan ver qué sistemas de archivos estamos compartiendo se requiere agregar, en dicho archivo, los sistemas de archivos a compartir y qué segmento de red o qué direcciones IP específicas podrán acceder a ellos.

```
zion0:/usr/local/sbin/admon# ./rc-ssh cat /etc/exports

***** zion0 *****
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).

/usr    192.168.0.0/255.255.255.0(rw,sync)
/tmp    192.168.0.0/255.255.255.0(rw,sync)

***** zion1 *****
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
/tmpu2    192.168.0.0/255.255.255.0(rw,sync)

***** zion2 *****
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
/home    192.168.0.0/255.255.255.0(rw,sync)

***** zion3 *****
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
/tmpu    192.168.0.0/255.255.255.0(rw,sync)
```

Capítulo 6. Seguridad

La seguridad es un campo complejo, en el cual se mezclan tanto técnicas de análisis, con otras de detección o de prevención de los posibles ataques, el comportamiento de los usuarios del sistema o las posibles intenciones de los atacantes [1].

Los ataques pueden provenir de muchas fuentes y afectar desde a una aplicación o servicio, hasta algún usuario o al sistema entero. Podemos encontrarnos con ataques de autenticación (obtener acceso por parte de programas o usuarios previamente no habilitados), sniffers (redirigir los canales de comunicación y los datos que circulan en una red), o sustitución (sustituir programas, máquinas o usuarios por otros, sin que se noten los cambios) entre otros.

Ante éstos y otros posibles ataques, tenemos que disponer de instrumentos normativos y mecanismos tecnológicos de prevención, detección y recuperación de nuestro sistema.

Dentro de los instrumentos normativos tenemos la definición de una serie de buenas prácticas entre las más importantes están:

6.1 Buenas prácticas

- El password de root es conocido sólo por el Jefe del Departamento, los administradores deberán contar con mecanismos que permitan ejecutar con privilegios las tareas de administración que así lo requieran.

- Se otorgan privilegios a los usuarios a medida que se lleguen a necesitar y previa autorización del Jefe de Departamento.

- La solicitud de una cuenta de usuario en el cluster estará determinada por el Jefe de Departamento.

- Si un usuario requiere trabajar en consola (acceso físico al cluster) deberá solicitar autorización del Jefe de Departamento.

- Para el acceso remoto al cluster, se deberá definir una IP específica que estará relacionada directamente con el usuario.

- Se revisarán periódicamente las actualizaciones de kernel, servicios y aplicaciones instaladas en el cluster. Para su aplicación se requiere previa autorización del Jefe de Departamento y una evaluación a detalle que verifique que dicha actualización no afectará ninguno de los servicios ofrecidos por el cluster, en caso de hacerlo, se valorarán alternativas que permitan la protección del cluster.

Para la aplicación de los mecanismos tecnológicos que nos permitan llevar a buen término la implantación de las buenas prácticas previamente definidas, se dividió la seguridad del sistema en dos grandes ramas: *la seguridad local y la seguridad en red*.

En cuanto a la **seguridad local**, debemos examinar los diferentes mecanismos de autenticación y permisos de accesos a los recursos para definirlos correctamente y poder garantizar la confidencialidad y la integridad de la información.

Respecto a la **seguridad en red**, debemos garantizar que los recursos que ofrecemos tienen los parámetros de confidencialidad necesarios y que los servicios no pueden ser usados por terceros no deseados, de modo que, un primer paso será controlar que los servicios ofrecidos sean los que realmente queremos, y que no estamos permitiendo servicios no controlados.

En el campo correspondiente a las aplicaciones y a los mismos servicios, además de garantizar la correcta configuración de niveles de acceso mediante permisos y de autenticación de los usuarios permitidos, tendremos que vigilar la posible explotación de bugs (errores) en el software. Cualquier aplicación, por muy bien diseñada e implementada que esté, puede tener un número más o menos alto de errores que pueden ser aprovechados para, con ciertas técnicas, evadir las restricciones impuestas. En este caso, y de acuerdo a las buenas prácticas previamente definidas, se mantendrá el sistema actualizado siempre y cuando dicha actualización no afecte el desempeño de alguna aplicación gráfica requerida. Así, la distribución de Linux seleccionada para operar con el Cluster –Debian- permite un manejo de actualizaciones y administración de paquetes bastante potente y confiable, siendo una de las características mas reconocidas la pronta corrección de errores y actualizaciones del sistema.

6.2 Seguridad Local

La seguridad local es básica para la protección del sistema, ya que normalmente, tras un primer intento de acceso desde la red, es la segunda barrera de protección antes de que un ataque se haga del control de la máquina.

Además, dadas las características del Cluster, la mayoría de las aplicaciones con que cuenta deben ser ejecutadas localmente, como por ejemplo, NetJuggler que permite hacer el despliegue de una imagen en varias pantallas.

De entrada, ya en el arranque del sistema se nos pueden presentar problemas debido al acceso físico que un intruso o usuario pueda tener a la máquina. Si el sistema puede arrancarse de disco o CD, un atacante podría acceder a los datos de una partición tan sólo con montar el sistema de archivos y podría colocarse como usuario root sin necesidad de conocer ninguna contraseña. En este caso, se necesita proteger el arranque del sistema desde el BIOS, protegiendo el acceso por contraseña, de manera que no se permita el arranque desde CD o disquete, siendo esta la primer medida de seguridad del Cluster.

El siguiente paso es proteger el *bootloader* o gestor de arranque, en el Cluster Zion, la distribución Debian incluye por default el gestor de arranque *lilo.* Para que el usuario local o el atacante no pueda modificar las opciones de arranque del *kernel* o modificar directamente el arranque se puede colocar una contraseña esto puede llevar a cabo de la siguiente manera siendo otra medida de seguridad para el Cluster.

En *lilo* se coloca una contraseña global en el archivo /etc/lilo.conf con:

`password = contraseña`

Hay que tener cuidado de poner el archivo /etc/lilo.conf protegido a solo lectura/escritura desde el *root* (`chmod 600 /etc/lilo.conf`).

Otro tema relacionado con el arranque es la posibilidad de que alguien que tenga acceso al teclado reinicie el sistema, pulsando las teclas CTRL+ALT+DEL, provocando una operación de *shutdown* en la máquina. Este comportamiento viene definido en /etc/inittab, con una línea como:

```
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
```

Para prevenir este ataque podemos hacerlo de dos maneras distintas. Una es comentando la línea anterior en el archivo /etc/inittab o bien, puede crearse un archivo /etc/shutdown.allow, que permite a ciertos usuarios poder reiniciar que es la solución implementada en el Cluster Zion. El archivo /etc/shutdown.allow es el siguiente:

```
zion0:~# cat /etc/shutdown.allow
cesar
marce
root
```

Siguiendo con el tema de la seguridad local, en el sistema existen unos cuantos archivos de configuración especiales que permiten habilitar el acceso a una serie de *hosts* a algunos servicios de red, pero cuyos errores pueden permitir atacar después la seguridad local como son .rhosts el /etc/hosts.equiv o bien /etc/hosts.lpd entre otros, lo mas recomendable y que fue implementado en el Cluster Zion es deshabilitar los servicios (como lpd) de que hacen uso y emplear conexiones encriptadas con SSH o SSL.

Además, como proveedores de algunos servicios, debemos asegurar que no pongamos en peligro a nuestros usuarios (o se pongan en peligro solos) por usar servicios inseguros. Evitando el uso de servicios que no utilicen encriptación de datos y contraseñas (ftp, telnet, correo no seguro) esto ya dentro de la seguridad en red.

6.3 Seguridad en red

6.3.1 Protección mediante filtrado. TCPwrappers.

TCP wrappers es un software que actúa de intermediario entre las peticiones del usuario de servicio y los demonios (*daemons*) de los servidores que ofrecen el servicio. Muchas de las distribuciones vienen ya con los *wrappers* activados y podemos configurar los niveles de acceso. Los *wrappers* se suelen utilizar en combinación con *inetd* o *xinetd*, de manera que protejan los servicios que ofrecen.

El *wrapper* básicamente sustituye al demonio del servicio por otro que actúa de intermediario (llamado *tcpd*, normalmente en /usr/sbin/tcpd). Cuando éste recibe una petición, verifica el usuario y origen de ésta, para determinar si la configuración del *wrapper* del servicio permite o no utilizarlo. Además, incorpora facilidades de generar *logs*, o bien informar por correo de los posibles intentos de acceso y después ejecuta el *daemon* adecuado asignado al servicio.

Existe un método alternativo de *wrapper* TCP, que consiste en que la aplicación original se compile con la biblioteca de *wrappers*. De esta manera, la aplicación no tiene por qué estar en *inetd*, y podremos controlarla igual que el primer caso con la configuración que comentamos a continuación y que es la empleada en el Cluster Zion.

El sistema de *wrappers* se controla donde los archivos /etc/hosts.deny, en que especificamos qué servicios denegamos y a quién, por medio de opciones, como un pequeño *shell* para guardar la información del intento, y el archivo /etc/hosts.allow, donde solemos colocar qué servicio vamos a utilizar, seguido de la lista de a quién dejamos entrar. Para nuestro caso, se ha implementado un script en Perl llamado alert.pl que descrito de manera general, envía un correo al administrador del Cluster si han habido tres intentos fallidos por hacer uso de

algún servicio. Los archivos empleados en la configuración de TCPwrappers se encuentran en el Anexo B.

6.3.2 Firewall

Un firewall es un sistema o grupo de sistemas que refuerza las políticas de control de acceso entre redes. El firewall puede estar implementado en software, como una aplicación especializada corriendo en una computadora individual, o bien puede tratarse de un dispositivo especial dedicado a proteger una o más computadoras [1].

Técnicamente, la mejor solución es disponer de una computadora con dos o más tarjetas de red que aíslen las diferentes redes (o segmentos de red) conectados, de manera que el software de firewall en la máquina se encargue de conectar los paquetes de las redes y determinar cuáles pueden pasar o no, y a qué red.

Otra configuración típica es la de firewall hacia Internet, por ejemplo con dos tarjetas de red: en una obtenemos/proporcionamos tráfico a Internet y en la otra enviamos o proporcionamos el tráfico a nuestra red interna, eliminando el tráfico que no va destinado a nosotros, y también controlar el tráfico que se mueve hacia Internet, por si no queremos que se tenga acceso a algunos protocolos, o bien sospechamos que hay posibilidades de fugas de información por algunos ataques.

El firewall en general permite definir al usuario una serie de políticas de acceso (cuáles son las máquinas a las que se puede conectar o las que pueden recibir información y el tipo de información) por medio del control de los puertos TCP/UDP permitidos de entrada (*incomming*) o de salida (*outcomming*). Otra técnica a veces relacionada es NAT (*Network Address Translation*). Esta técnica

proporciona una vía para ocultar las direcciones IP usadas en la red privada y las oculta de Internet, pero mantiene el acceso desde las máquinas. Uno de los métodos típicos es el denominado *masquerading*. Utilizando *NAT masquerading*, uno o varios dispositivos en la red pueden aparecer como una única dirección IP vistos desde fuera.

En el caso especifico del Cluster Zion, el firewall implementado, utiliza cada una de las técnicas antes mencionadas, es decir:

Protegerá una máquina que se encuentra en Internet que es el nodo maestro Zion0. Servirá para crear dos redes una interna para el resto de los nodos del cluster y otra externa que conforma el nodo cero para acceso al cluster.

Protegerá la red interna, es decir, dará protección a los nodos internos de ataques del exterior y finalmente se creara un esquema de NAT para el proceso de instalación y actualización del cluster ya que los nodos requerirán de acceso a Internet para descargar ciertas aplicaciones y sus actualizaciones.

Por las características específicas del Cluster se decidió crear un script en bash llamado firewall que se colocará en los niveles de arranque (runlevels) bajo el nombre de **/etc/init.d/firewall**. Implementado con una política restrictiva , es decir:

Todo está prohibido a menos que se encuentre previamente permitido.

El script empleado en el Cluster y que se encuentra bajo /etc/init.d/firewall puede verse en el Anexo B.

6.4 Seguridad en la Administración

6.4.1 Protección de la cuenta root.

Para evitar el compromiso del password de root y como una buena práctica de administración, se configuró la utilería sudo para asignar a algunos usuarios privilegios de súper-usuario en orden de facilitar la administración del cluster.

Sudo es una herramienta que permite al administrador de un sistema Linux dar a ciertos usuarios (o grupo de usuarios) la habilidad de ejecutar algunos (o todos) comandos como root y mandar a bitácoras dicha actividad. Su filosofía es dar a los usuarios la menor cantidad de privilegios posibles pero siempre permitiéndoles hacer su trabajo. Dentro de sus características tenemos:

La habilidad para restringir qué comandos puede un usuario ejecutar en el sistema.

Envía a bitácoras toda la actividad relacionada con la ejecución de comandos.

Emplea marcas de tiempo que pueden ser configurables es decir, cuando un usuario ejecuta un comando a través de sudo, éste solicita su password el cual tiene un tiempo de vida de 5 minutos durante los cuales no le solicitará nuevamente su password, una vez expirado dicho tiempo se debe volver a teclear el password del usuario.

La aplicación sudo no es instalada por default en los sistemas UNIX-Linux, pero está disponible para la mayoría de ellos, su instalación en Debian es bastante sencilla.

```
#apt-get install sudo
```

Mediante el comando visudo se edita el archivo /etc/sudores. Este archivo, contiene macros que definen el nombre del host, los comandos a ejecutar y los usuarios permitidos para ejecutar dichos comandos en el equipo.

Por default, visudo abre el archivo utilizando el editor de texto pico, para abrirlo con el editor de texto preferido, borrar la liga que se encuentra en /etc/alternatives/editor y crear una nueva apuntando a vi.

```
#rm /etc/alternatives/editor
#ln –s /usr/bin/vi /etc/alternatives/editor
```

Así, el archivo de configuración de la aplicación sudo es /etc/sudores y puede verse en el Anexo B.

Capítulo 7. Monitoreo

El monitoreo de cualquier sistema es una tarea vital dentro de la administración de éste, teniendo un buen análisis y monitoreo de nuestro sistema podemos darnos cuenta de aspectos como la seguridad, que tan bueno es su desempeño (procesos, memoria, uso de cpu, etc.) e incluso saber que tan bien administrado se encuentra.

Para el caso del cluster Zion, se definieron dos aspectos importantes que requieren de un proceso de análisis y monitoreo: uno de ellos son la bitácoras del sistema y el otro no menos importante el desempeño del mismo.

7.1 Análisis de bitácoras (logs)

Dando una visión sobre los ficheros de *log,* podemos darnos una idea rápida del estado global del sistema, así como de las últimas ocurrencias, y detectar accesos (o intentos) indebidos.

Hay que tener cuidado con los *logs* del sistema, si ha habido una intrusión real, pueden haber sido limpiados o alterados. La mayoría de archivos de *log* residen en el directorio /var/log.

Muchos de los servicios pueden poseer *logs* propios, que normalmente se establecen en la configuración (mediante el correspondiente archivo de configuración). La mayoría suelen utilizar las facilidades de *log* incorporadas en el sistema Syslog, mediante el demonio Syslogd. Su configuración reside en /etc./syslog.conf. Esta configuración suele establecerse por niveles de mensajes: existen diferentes tipos de mensajes según su importancia. Normalmente suelen aparecer niveles *debug, info, err, notice, warning, err, crit, alert, emerg*, más o menos éste es el orden de importancia de los mensajes (de menor a mayor). Generalmente, la mayoría de los mensajes se dirigen al *log* /var/log/messages, pero puede definirse que cada tipo vaya a archivos diferentes y también se puede identificar quién los ha originado; típicamente el *kernel*, correo, *news*, el sistema de autentificación, etc [1].

Por lo tanto, hay que examinar y adaptar la configuración de Syslog para conocer en qué *logs* podemos encontrar/generar la información. Otro punto importante es controlar su crecimiento, ya que según los que estén activos y las operaciones (y servicios) que se realicen en el sistema, los *logs* pueden crecer bastante. En Debian se controla a través de *logrotated*, un demonio que se encarga periódicamente de hacer copias de los logs más antiguos y comprimirlas, se puede encontrar su configuración general en /etc./logrotate.conf, algunas aplicaciones hacen configuraciones específicas que se pueden encontrar en el directorio /etc./logrotate.d.

Así pues, las bitácoras a analizar en cluster Zion son las que el sistema operativo en nuestro caso Debian trae por default, entre ellas las más importantes son las siguientes:

- /var/log/messages: Es el archivo de *log* por defecto del demonio Syslogd, pero habría que revisar su configuración. Este fichero contiene una amplia variedad de mensajes de orígenes diversos (diferentes demonios, servicios, la opción

de log de iptables o el mismo *kernel*); todo lo que se observase anómalo tendría que ser verificado.

- /var/log/utmp: Este archivo contiene información binaria para cada usuario que está actualmente activo. Es interesante para determinar quién está dentro del sistema. El comando *who* utiliza este fichero para proporcionar esta información.

- /var/log/wtmp: Cada vez que un usuario entra en el sistema, sale, o la máquina reinicia, se guarda una entrada en este archivo. Es un archivo binario de donde el comando *last* obtiene información, en que se menciona qué usuarios entraron o salieron, cuándo y dónde se originó la conexión. Puede ser útil para buscar dónde (en qué cuentas) se originó la intrusión o detectar usos de cuentas sospechosas. También existe una variación del comando llamada *lastb*, que lista los intentos de *login* que no pudieron validarse correctamente y se usa el archivo /var/log/btmp (puede ser necesario crearlo si no existe). Estos mismos fallos de autentificación también suelen enviarse al *log auth.log*. De modo parecido, el comando *lastlog* utiliza otro archivo /var/loglastlog para verificar cuál fue la última conexión de cada uno de los usuarios.

- /var/log/secure: Pueden buscarse intentos de intrusos de servicios que no suelen usarse, o bien máquinas no familiares que se intentan conectar.

- /var/log/httpd: Si tenemos servidor web, aquí se logean nuestras conexiones.

- maillog: Envía a bitácoras todos los mails a un archivo

- cron: Envía a bitácoras el cron /var/log/cron

- boot.log : Envía a bitácoras los mensajes de arranque del sistema

En el sistema de *logs*, otra cosa a tener en cuenta es que el directorio de logs /var/log, pueda ser modificado sólo por *root*. De otro modo, cualquier atacante podría falsificar la información de los *logs.* Por ejemplo, si el atacante consigue acceso a *root,* puede, y suele, borrar las pista de sus accesos. Para asegurar de alguna manera que la información de los logs del sistema sea íntegra, se cuentan con varias técnicas para respaldar dichos logs, la primera es la creación de un servidor de logs que se encuentre en un maquina diferente es decir, enviar los logs o bitácoras a un servidor centralizado que sirva de repositorio, esto es bastante sencillo de implementar modificando el archivo /etc./syslog.conf sin embargo se requiere de un servidor con suficientes recursos. La segunda técnica y que fue implementada en el Cluster Zion, es la generación de reportes periódicos de los logs del sistema, y ver que cambios o alertas vemos en ellas, esto se realiza mediante logwatch, una utilería que permite la generación de reportes de los logs del sistemas para su mejor comprensión y pueden ser calendarizadas para enviar el reporte por correo electrónico.

Una de las características principales de logwatch es que permite generar reportes con diferentes niveles de análisis, es decir, se cuenta con un análisis básico, medio y a profundidad de los logs del sistema.

El esquema de reportes de logs mediante logwatch en el Cluster Zion se encuentra establecido a la generación de un reporte sencillo y enviando diariamente un reporte al administrador del sistema esto se indica en la configuración de logwatch en el archivo:
/etc/logwatch/conf/logwatch.conf.

En Anexo B se muestra un correo generado por logwatch enviado al administrador del sistema.

7.2 Monitoreo del desempeño del sistema (Ganglia).

Todo cluster requiere de un análisis de rendimiento que entre otras cosas nos indique la carga del sistema, el tiempo de uso de cpu, la carga de la red interna, procesos etc. Sin embargo, este punto se complica al tratarse de un cluster o conjunto de máquinas ya que la mayoría de las veces se cuenta con métricas de rendimiento individuales, es decir, de cada uno de los nodos que conforman al cluster por separado, haciendo un poco mas difícil al administrador tener una idea global del comportamiento general del cluster en tiempo real.

7.2.1 Ganglia

Ganglia es un sistema de monitoreo en tiempo real que provee datos estadísticos y métricas de desempeño para sistemas basados en tecnología cluster. Una de sus principales características es que emplea herramientas libres y populares como son RRDtools, XRD, XML y PHP para su operación. Las métricas que analiza ganglia son [8]:

```
  Metric Name    Description
Platforms
  ----------------------------------------------------------------
-------
  boottime      System boot timestamp                    l,f
  bread_sec
  bwrite_sec
  bytes_in      Number of bytes in per second            l,f
  bytes_out     Number of bytes out per second           l,f
  cpu_aidle     Percent of time since boot idle CPU      l
  cpu_arm
  cpu_avm
  cpu_idle      Percent CPU idle                         l,f
  cpu_intr
  cpu_nice      Percent CPU nice                         l,f
  cpu_num       Number of CPUs                           l,f
  cpu_rm
```

```
cpu_speed     Speed in MHz of CPU                      l,f
cpu_ssys
cpu_system    Percent CPU system                       l,f
cpu_user      Percent CPU user                         l,f
cpu_vm
cpu_wait
cpu_wio
disk_free     Total free disk space                    l,f
disk_total    Total available disk space               l,f
load_fifteen  Fifteen minute load average              l,f
load_five     Five minute load average                 l,f
load_one      One minute load average                  l,f
location      GPS coordinates for host                 e
lread_sec
lwrite_sec
machine_type
mem_buffers   Amount of buffered memory                l,f
mem_cached    Amount of cached memory                  l,f
mem_free      Amount of available memory               l,f
mem_shared    Amount of shared memory                  l,f
mem_total     Amount of available memory               l,f
mtu           Network maximum transmission unit        l,f
os_name       Operating system name                    l,f
os_release    Operating system release (version)       l,f
part_max_used Maximum percent used for all partitions  l,f
phread_sec
phwrite_sec
pkts_in       Packets in per second                    l,f
pkts_out      Packets out per second                   l,f
proc_run      Total number of running processes        l,f
proc_total    Total number of processes                l,f
rcache
swap_free     Amount of available swap memory          l,f
swap_total    Total amount of swap memory              l,f
sys_clock     Current time on host                     l,f
wcache
Platform key:
l = Linux, f = FreeBSD, a = AIX, c = Cygwin
m = MacOS, i = IRIX, h = HPUX,  t = Tru64
e = Every Platform
```

El sistema de monitoreo ganglia está conformado de tres componentes principales:

El demonio de monitoreo GMOND. Que corre en cada nodo que va a ser monitoreado.

El meta demonio de ganglia GMETAD que se ejecuta en el nodo principal o aquel que contendrá el servidor web donde se desplegaran los datos.

El Front-End de ganglia que es el encargado de ejecutar la página web donde se despliegan los datos del monitoreo total del sistema.

Algunos puntos importantes a considerar durante la instalación de ganglia son:

- En un sistema Cluster, sólo un nodo deberá ejecutar el demonio GMETAD. El resto de los nodos ejecutarán solamente el demonio GMOND. El nodo que ejecute el demonio GMETAD deberá ejecutar el demonio GMOND sólo si se desea incluir dentro del monitoreo. En el cluster Zion, el nodo principal Zion0 se encuentra ejecutando ambos demonios, mientras que el resto de los nodos sólo ejecutan el demonio GMOND.
- La máquina que ejecute el demonio GMETAD será la encargada de agregar y representar los datos del resto de los nodos a monitorear por lo que requiere de las utilerías RRDtools. Para nuestro caso, el nodo Zion0 realizará esta función.
- Deberá haber un nodo que actúe como servidor Web siendo necesario que GMETAD sea ejecutado en dicha máquina.
- La máquina que actúe como servidor Web deberá tener instalado PHP.

7.2.2 Instalación de Ganglia.

El proceso de instalación de Ganglia es bastante simple ya que se cuenta con paquete Debian para su instalación, previamente se deberá instalar el servidor Web Apache con soporte para PHP, así los pasos a seguir son:

Instalar RRDtools.

```
apt-get install rrdtool
```

Instalación de ganglia:

```
apt-get install ganglia
```

Dado que la utilería en su totalidad se instala bajo el directorio /usr y recordando que dicho sistema de archivos se encuentra exportado a todos y cada uno de los nodos que componen nuestro cluster, lo que resta es hacer una copia del script de inicialización del demonio en cada uno de los nodos a monitorear. Así pues, hacemos la copia de dicho script al directorio /etc./init.d/.

La dirección para verificar que ganglia se encuentre en funcionamiento es:

http://localhost/ganglia/

7.2.3 Configuración.

7.2.3.1 Configuración del demonio GMOND

El demonio GMOND por default trabaja en la mayoría de los clusters sin embargo, se sugiere hacer algunos cambios al archivo de configuración /etc./gmond.conf. Algunas de las variables más importantes a modificar son [8]:

name. Asignar a esta variable el nombre de nuestro cluster

mcast_channel. Existe un canal por default que GMOND utiliza para comunicarse con el servidor principal empleando la dirección multicast 239.2.11.71. No debe ser cambiado a menos que exista alguna otra aplicación que este utilizando el mismo canal.

mcast_port. El Puerto multicast mcast_port especifica el puerto por donde se comunicará GMOND al igual que el canal multicast no debe cambiarse a menos que ya se encuentre en uso por alguna otra aplicación.

mcast_if. Este valor es asignado por GMOND basado en información proporcionada por el kernel. Sin embargo, se presenta el caso en que existen múltiples interfaces de red. Para el caso del nodo principal este valor es puesto a eth1 mientras que para el resto de los nodos será eth0.

num_nodes. Esta variable es utilizada para especificar el número de nodos del cluster al que pertenece la máquina. Se recomienda dejar el valor por default.

Estas son las variables más representativas e importantes a tomar en cuenta durante la configuración del demonio GMOND. Ahora se puede reiniciar el demonio y verificar que se encuentre en ejecución.

7.2.3.2 Configuración del demonio GMETAD

Recordemos que el demonio GMETAD sólo se encuentra instalado en el nodo maestro de nuestro cluster así pues, a diferencia del demonio GMOND, la configuración de este demonio es única del nodo Zion0.

data_source. La variable data_source es la más importante en la configuración y es imperativo que sea asignado el valor correcto del nombre del cluster que será monitoreado para que el sistema pueda monitorear el estado de dicho cluster.

xml_port. GMETAD envía sus datos a través del puerto 8651 por default, se recomienda no modificar esta opción.

rrd_rootdir. Se menciono durante la instalación de RRDtools que esta utilería es instala en el directorio por default. Esta opción deberá apuntar al directorio donde se encuentra instalada RRDtools para nuestro caso /var/lib/ganglia/rrds/.

Así, tenemos configurado completamente nuestro sistema de monitoreo Ganglia y podemos comenzar a monitorear el comportamiento y desempeño de nuestro cluster. Algunas capturas de pantalla de la aplicación se muestran a continuación:

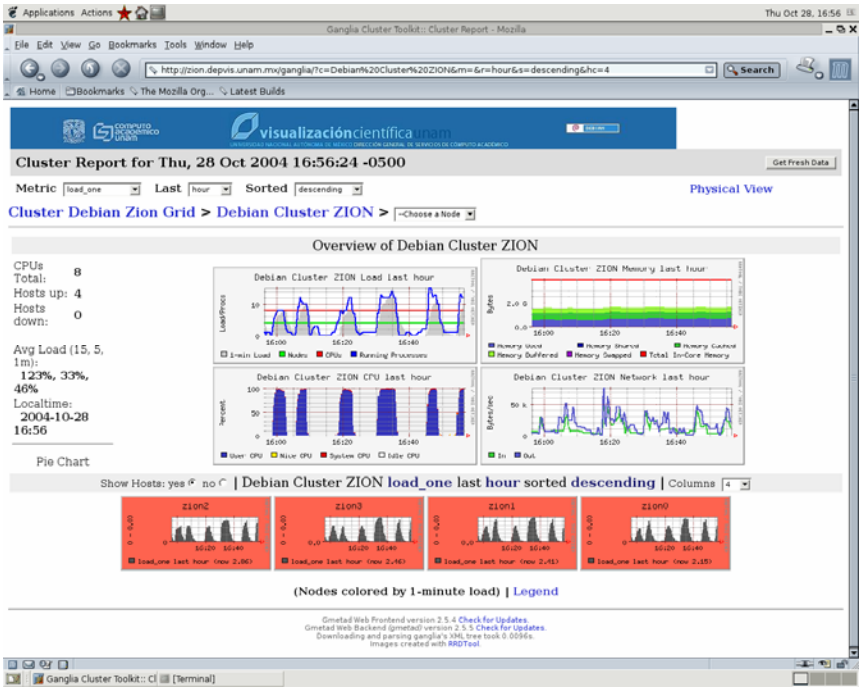

Figura 7.1 Tomado durante la ejecución de un benchmarck de rendimiento del cluster, se pude observar claramente la carga del sistema con un cambio de color en c/u de los nodos.

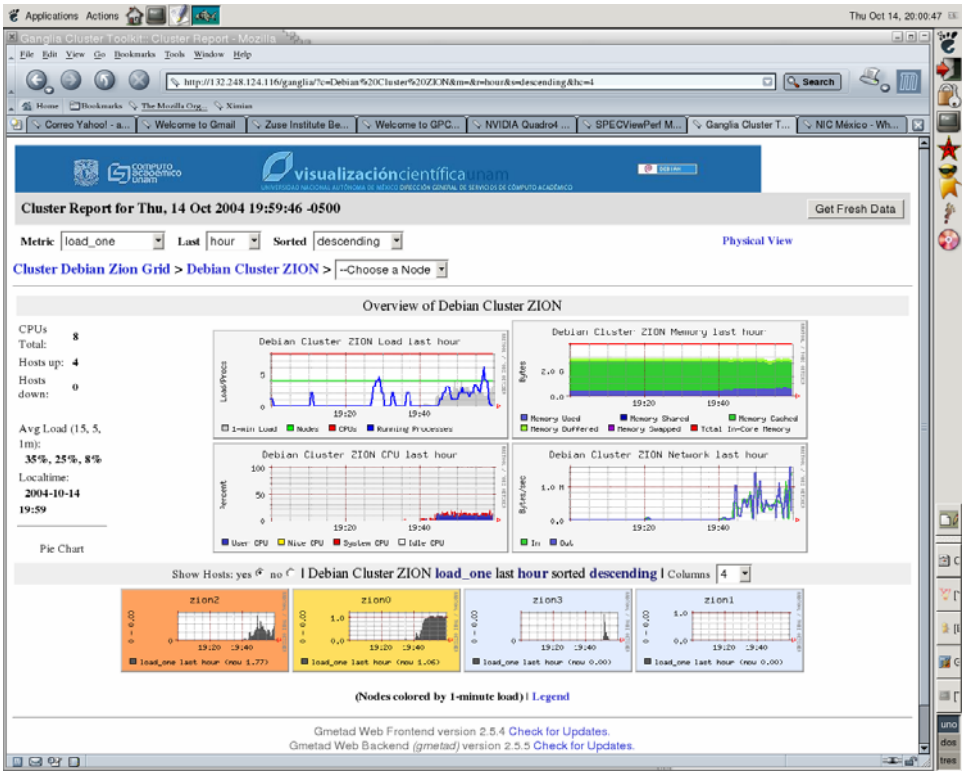

Figura 7.2. Tomado durante la ejecución de diferentes procesos en cada uno de los nodos.

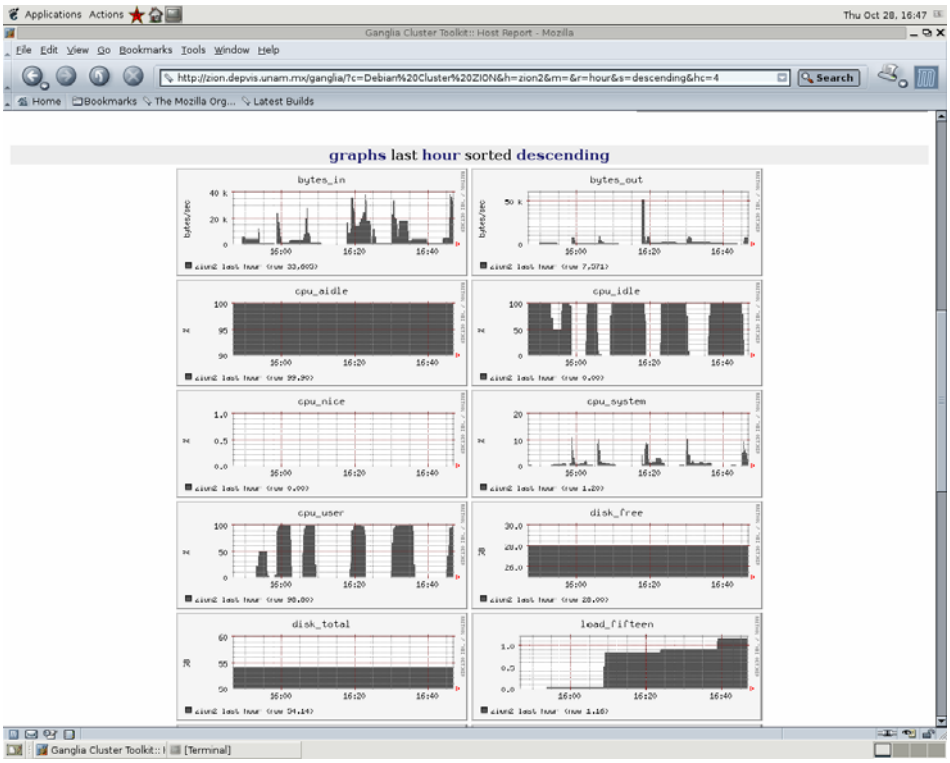

Figura 7.3. Vista de algunas de las métricas tomadas por Ganglia.

Capítulo 8. Aplicaciones gráficas instaladas y ejemplos.

A continuación se presenta el proceso de instalación, algunos ejemplos, y problemas presentados durante la instalación de las aplicaciones gráficas que se encuentran instaladas en el Cluster Zion.

8.1 vr++ 0.7.1 1

Es una herramienta open source que permite crear aplicaciones modulares de realidad virtual. Los desarrollos en VR++ consisten en múltiples tareas que corren paralelamente y que se comunican unas con otras a través de una interfaz estándar de datos, lo cual permite que todas las aplicaciones sean compatibles unas con otras. VR++ usa el esquema maestro - esclavo, para coordinar el trabajo entre los múltiples procesos, además tiene varios esquemas de comunicación entre los procesos (maestro- esclavo, esclavo - esclavo) [16].

Instalación

- `make user.make`
- `Defina en user.make valores a MPI CXX = mpic++, MPI INC = -I/usr/include/lam/, MPI LIBS = -L/usr/lib/lam/`
- `make`
- `make docs`

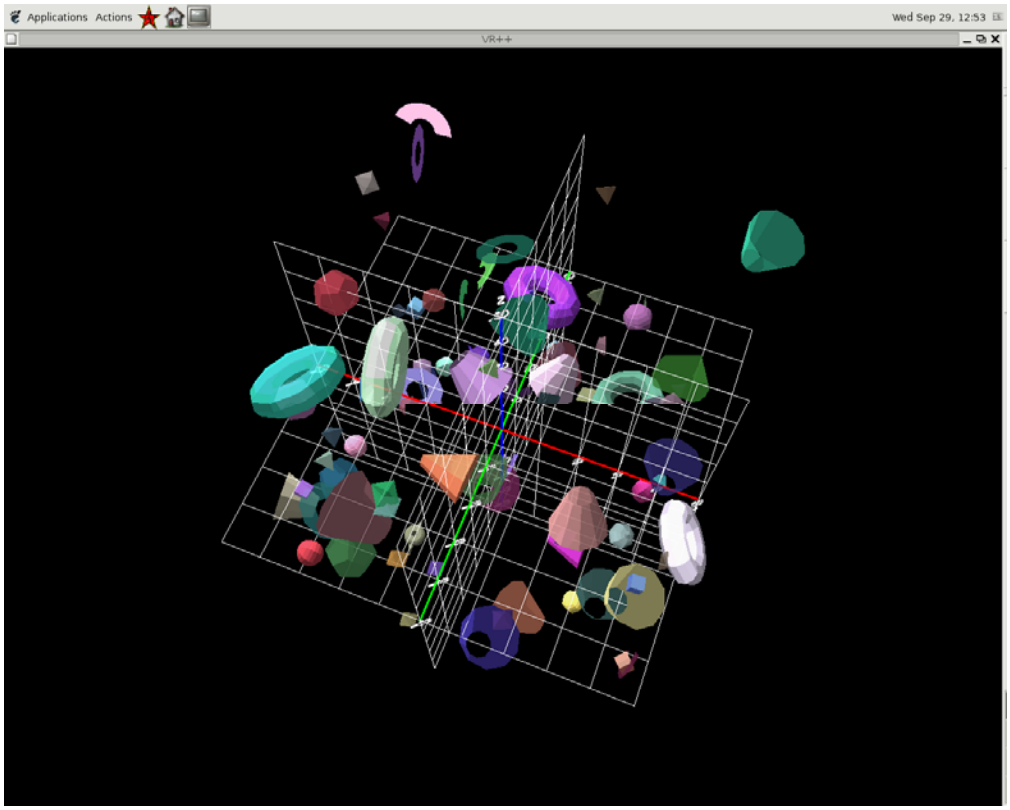

Figura 8.1. Ejemplo de vr++

8.2 Performer

Performer es una poderosa e intuitiva interfaz de programación para el desarrollo de aplicaciones de simulación visual en tiempo real y otros gráficos 3D de alto desempeño.

El conjunto de herramientas simplifica el desarrollo de aplicaciones usadas para simulaciones visuales, diseño basado en simulación, realidad virtual, visualización científica, entretenimiento interactivo, video, diseño asistido por computadora, etc.

Performer está basado en el estándar de la librería gráfica OpenGL, permite interoperar con OpenGL Volumizer, OpenGL Multipipe SDK, y OpenGL

Vizserver. Esta disponible para Irix, Linux, Windows XP y Windows 2000 [17].

La instalación es bastante sencilla, se tratan de binarios que sólo deben ser desempaquetados. La forma de ejecución de ejemplos es: `/usr/bin/X11/perfly archivo.pfb`

8.3 Vrjuggler

Es un proyecto Open Source iniciado en 1997 en la Universidad Estatal de Iowa. Soporta múltiples sistemas, guantes y dispositivos de entrada y puede controlar casi cualquier configuración de despliegue. Hasta este momento, VR Juggler ha sido portado a numerosas plaaformas como IRIX, Windows, Solaris y Linux [18].

Instalación:

- `./configure -prefix=/usr/local/vrjuggler -wth-gcc -with-jdkhome=/usr/local/j2sdk1.4.2_04/ -with-pfroot=/usr/`
- `make`
- `make install`
- `Agregar variable de ambiente VJ BASE DIR=/usr/local/vrjuggler`

`NOTA: esta compilado con g++-3.0.4 además de usar performer 3.1`

8.4 Netjuggler

Simula una sola máquina de Vrjuggler a partir de varias que lo ejecutan de manera independiente. En otras palabras permite ejecutar una aplicación de VR Juggler de igual manera en una PC que en un Cluster o una Onyx de SGI sin notar diferencias (desde el punto de vista operacional. y no desde el punto de vista de desempeño) [19].

Vrjuggler 1.07 para Netjuggler

Instalación

- tar -xvf vrjuggler-1.07.src.tar.gz
- mv vrjuggler-1.07.src vrjuggler
- cd vrjuggler
- patch -u -p 2 -i ../netjuggler/patch/vrjuggler-1.07.patch
- autoheader
- autoconf
- ./configure -prefix=/usr/local/Net-vrjuggler -with-gcc -with-jdkhome=/usr/local/j2sdk1.4.2_0 -with-pfroot=/usr/
- make
- make install

  Ha quedado instalado. Ahora se ejecutarán los ejemplos.

  Agregar variable de ambiente VJ BASE DIR=/usr/local/vrjuggler
- export LD LIBRARY PATH=$LD LIBRARY PATH:$VJ BASE DIR/lib
- cd $VJ BASE DIR/samples/ogl/cubes
- make

  Ejecución de ejemplos
- ./cubes $VJ BASE DIR/share/Data/configFiles/simstandalone.config

Netjuggler

Instalación:

- ./configure -prefix=/usr/local/Netjuggler -with-vj base dir=/usr/local/Net- vrjuggler -with-njrun=no
- make
- make install

  Agregar variable de ambiente NJ BASE DIR=/usr/local/Net-vrjuggler
- agregar variable de ambiente USE NETJUGGLER=yes

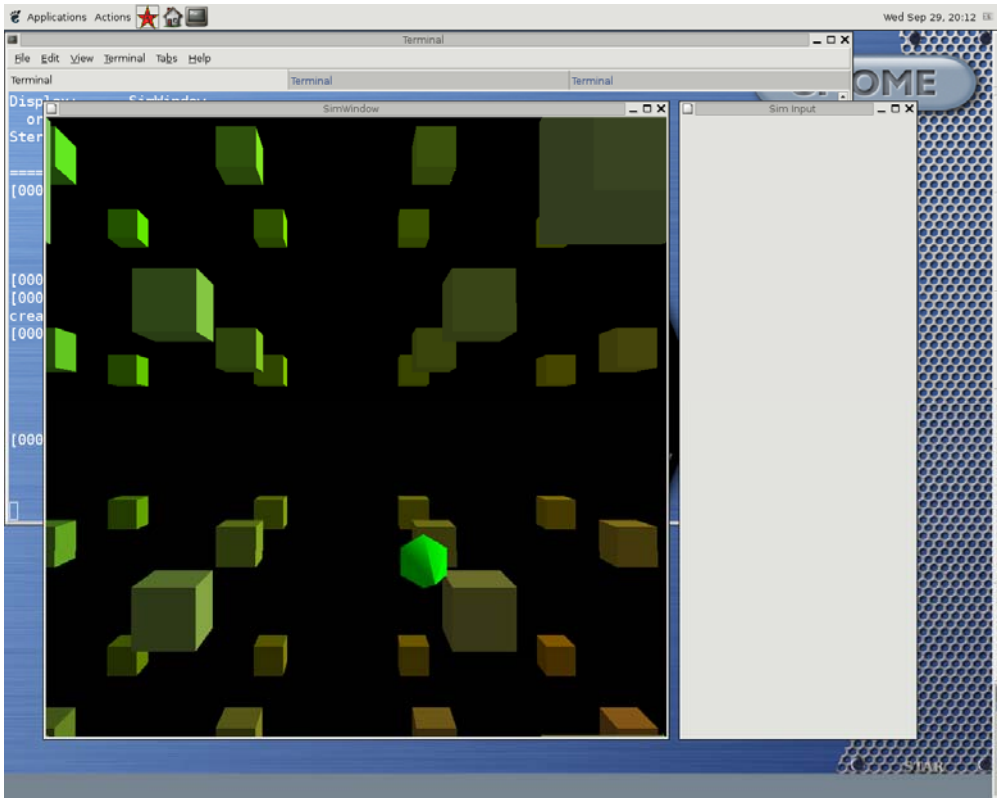

Figura 8.2. Ejemplo de Netjuggler

8.5 VTK

Es una herramienta open source para crear gráficos en 3D, procesamiento y visualización de imágenes. Consiste en bibliotecas escritas en C++, algunas de ellas también en Tcl/Tk, Java y Python. VTK soporta una amplia variedad de algoritmos de visualización (escalares, vectores, texturas e incluyendo métodos volumétricos), asi como técnicas avanzadas de modelación (modelado implícito, reducción por polígonos, mesh smooothing, triangulación Delaunay, cortes, contornos) [20].

Instalación:

- `apt-get install vtk-tcl vtkdata vtk-examples vtk-doc libvtk4 libvtk4-dev`

  Ejecución de ejemplos
- `export CXX=c++ compiler CC=c compiler`

- `mkdir vtk-arch`
- `cd vtk-arch`
- `ccmake -i vtk source dir`
- `mMake`
- `make install`

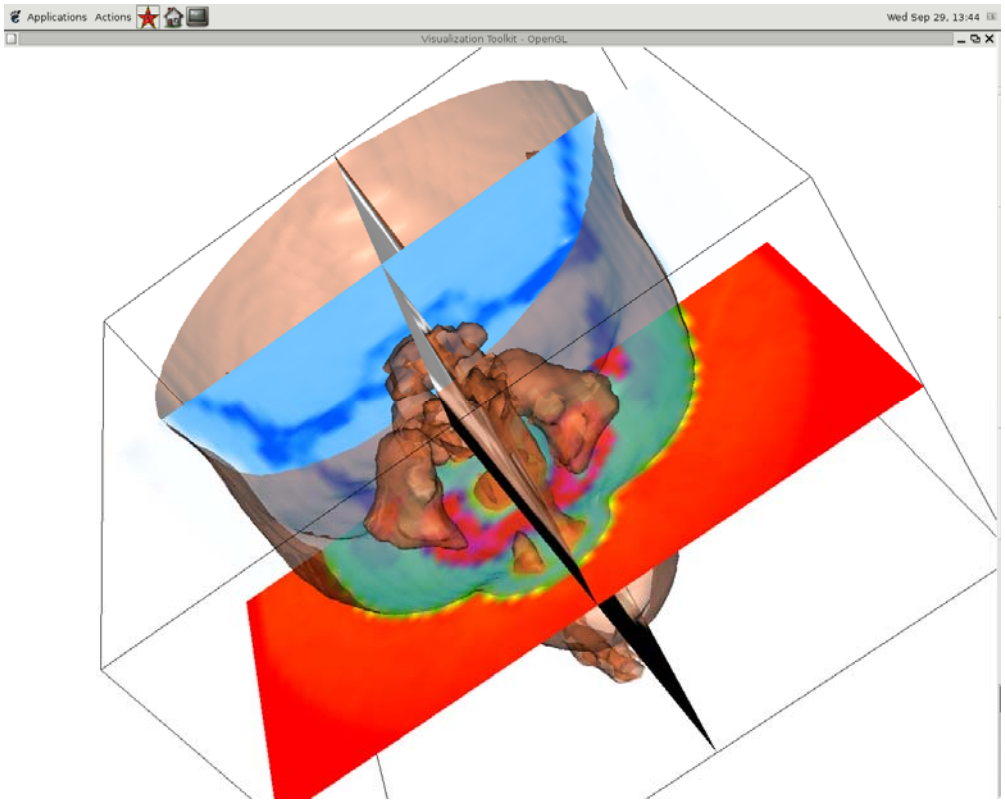

Figura 8.3. Ejemplo de VTK

8.6 Paraview

Es una aplicación surgida de la necesidad de visualizar grandes cantidades de datos. Paraview corre en memoria compartida y distribuida al igual que en sistemas con un solo procesador y ha sido probada en Windows, Linux y diversas estaciones de trabajo UNIX y clusters. Utiliza el Toolkit de Visualización (VTK) como la máquina de renderizado y procesamiento de datos

y tiene una interfaz de usuario escrita empleando una mezcla de Tcl/Tk y C++ [21].

Las principales metas de Paraview son:

- Desarollar una aplicación de visualización multiplataforma open source.
- Soportar modelos computacionales distribuidos para procesar grandes cantidades de datos.
- Crear una interfaz de usuario flexible e intuitiva.

La instalación es bastante sencilla, se trata de binarios que sólo deben ser descompactados y colocados bajo /usr/local

8.7 POVRAY

Es una herramienta open source para crear gráficos tridimensionales. Esta disponible en versiones oficiales para Windows, Macintosh y Linux. Su ubicación default es /usr/local [22].

Su instalación fue utilizando apt-get. El comando fue

```
apt-get install povray*
```

Lo que instala los siguientes paquetes:

```
povray - Dummy package depending on any concrete POV-Ray package
povray-3.5 - Persistence of vision raytracer (3D renderer)
povray-3.5-doc - Persistence of vision raytracer (3D renderer) documentation
povray-3.5-examples - Persistence of vision raytracer (3D renderer) sample files
povray-3.5-includes - Persistance of vision raytracer (3D renderer) include files
```

8.8 OpenDX

Es la primera opción para tareas de visualización desde crear imágenes visuales. hasta la construcción de aplicaciones completas. Es un poderoso software para visuallización científica, ingenieía y análisis de datos. Este sistema open source está construido sobre ambientes e interfaces estándares y su sofisticado modelo de datos provee a los usuarios de una gran flexibilidad en la visualización de datos [23].

Instalación:

- `./configure -prefix=/usr/local/dx-4.3.2 -enable-shared=yes -enable-smp-linux=yes -with-jni-path=/usr/local/j2sdk1.4.2i 04/include/ -with-rsh=/usr/bin/rsh`
- `make`
- `make install`
-

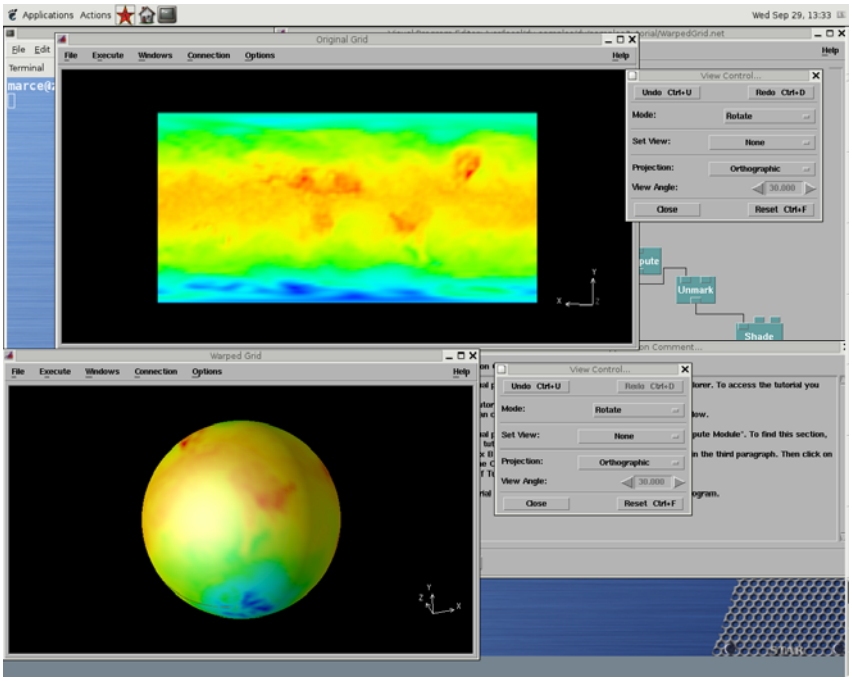

Figura 8.4. Ejemplo de OpenDX

8.9 OpenScenegraph-0.9.6-2

Conjunto de herramientas de alto desempeño para gráficos en 3D. Empleada para simulación visual, juegos, realidad virtual, visualización científica y modelado. Escrita totalmente en C++ y OpenGL, corre en las plataformas windows, OSX, Linux, IRIX, Solaris y FreeBSD [24].

Instalado en la ruta por default /usr/local/. Los fuentes están en `/usr/local/src/OSG OP OT- 0.9.6-2/`.

Debe definir las variables de ambiente :

- `LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/OSGFILE`
- `PATH=/usr/local/OpenSceneGraph-Data:/usr/local/OpenSceneGraph-Data/Images:/usr/local/OpenSceneGraph-Data/fonts`
- `PATH=$PATH:/usr/local/share/OpenSceneGraph`

8.10 VrNav2-5.8

Es un programa que permite la navegación en escenas durante la visualización de modelos 3D en ambientes de realidad virtual. Dentro de sus principales características se encuentran: permite cargar modelos con los formatos de datos soportados por OpenGL Performer, manejo de dispositivos tal como InterSence Tracker, joystick, teclados y mouse y soporta despliegues en estéreo y multipipe [25].

Instalación

- `Descargar y descompactar el paquete de vrNav2-5.8.tar.gz en: /usr/local/vrNav2-5.8`
- `Cambiarse sal directorio /usr/local/vrNav2-5./src/common.mk y modificar la ruta donde fue instalado vrjuggler.`
- `make`

- Establecer las rutas correctas de VRNAV HOME y VJ BASE DIR en el archivo ./ENVIRONS. Se puso lo siguiente: VJ_BASE_DIR=/home/marce/temporal/vrjuggler-2.0-alpha3.src/instlink
- Probar corriendo los ejemplos encontrados en el subdirectorio examples instalado en /usr/local/vrNav2-5.8

8.11 Chromium

Es un sistema para renderizado interactivo en clusters gráficos. Pueden implementarse varias técnicas de render paralelo como son: sort-fist y sort-last. Además, corre en Windows, IRIX y Linux.

Compilado con gcc-3.3 y su instalación necesita cambiar THREAD-on olo SAFE=1 en options.mk y gmake. Los binarios creados quedan dentro del mismo directorio de los fuentes; después fueron copiados a /usr/local/cr-1.7 [26].

8.12 OpenSG

Es un sitema portable de scenegraph para la creación de programas gráficos en tiempo real basado en OpenGL. Está desarrollado bajo los prinicpios del Open Source. Actualmente corre en Irix, Windows, y Linux. No es un sistema completo de realidad virtual, es la base de renderizado sobre la que son construidos sistemas de realidad virtual [27].

Instalación:

- cvs -d:pserver:anonymouscvs.sourceforge.net:/cvsroot/opensg login
- cvs -z3 -d:pserver:anonymouscvs.sourceforge.net:/cvsroot/opensg co OpenSG
- export CXX=g++-3.2
- ./configure -enable-glut -with-glutincludedir=/usr/include/GL -with-glutlibdir=/usr/lib -enable-qt -with-qt=/usr -with-qtincludedir=/usr/include -with-qtlibdir=/usr/lib -with-qtbindir=/usr/bin -enable-tif -enable-png -with-gif -enable-gif -

prefix=/usr/local/OpenSG-1.2.0-cvs

- make
- make install

Compilación de ejemplos:

- export LD LIBRARY PATH=$LD LIBRARY PATH:/usr/local/OpenSG- 1.2.0-cvs/lib/dbg
- cd $OpenSG Source Root/openSG/Tutorials
- make
- ejecución de los ejemplos
- Desde la máquina ollin se copiaron Xlib.h, Xutil.h, glu.h, gl.h
- Se instalaron manualmente archivos y/o ligas para paquetes glut-dev y gl- dev

NOTA:
OpenSG-1.2.0 estable compilado con g++ 3.0 no funciona, porque no per- mite compilar ejemplos
OpenSG-1.2.0 estable compilado con g++ 3.3 no funciona, porque no per- mite compilar ejemplos
OpenSG-1.2.0 version cvs compilado con g++ 3.2 funciona, permite compilar ejemplos, pero marca un error mientras es la ejecución de cubes

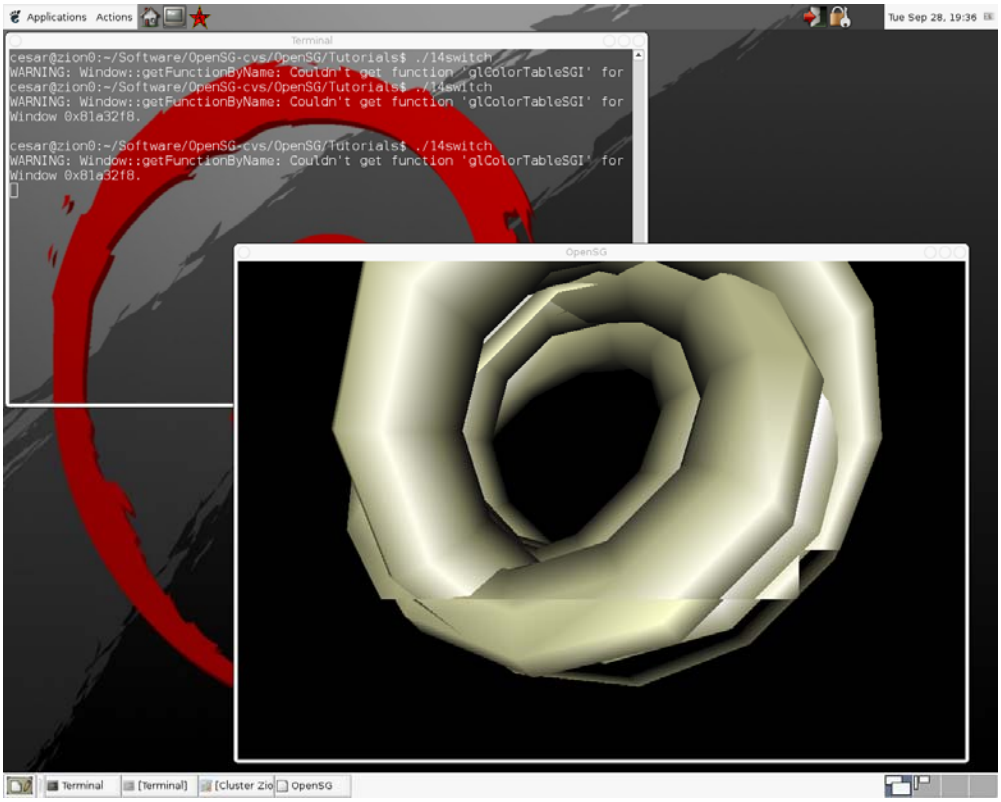

Figura 8.4. Ejejmplo de OpenSG

8.13 Syzysgy

La meta principal del proyecto Syzygy es ofrecer una herramienta para la ejecución de un amplia gama de aplicaciones de realidad virtual en cluster de PC's con igual o mejor desempeño que en sistemas de cómputo SMP de memoria compartida.

Syzygy incluye dos esquemas base: el esquema de grafo distribuido de escena para renderizar una única base de datos gráfica a múltiples clientes de render y un esquema mestro-esclavo con múltiples instancias de sincronización para las aplicaciones.

Sysygy incluye un OS distribuido simple y soporte para dispositivos de res, sound renders y renders gráficos, todos inmersos en su propia capa de red [28].

Instalación:

- `tar -xzvf szg-0.61.tar.gz`
- `modificar szg/build/make/Makefile.defines para evitar posibles errores conel ligador del compilador`
- `cd ./szg/build`
- `make`
- `make install`

8.14 OpenInventor

Es un conjunto de herramientas 3D orientadas a objetos que ofrecen una solución a problemas de programación de gráficos interactivos, ya que esta basado el modelo de programación en base de datos de escenas 3D. Incluye una amplia gama de objetos tales como cubos, polígonos, textos, cámaras,

luces, texturas, trackball, visualizadores 3D y editores [29].

Instalación

- `mkdir /usr/local/Inventor/`
- `cd /usr/local/Inventor/`
- `tar xzvf sgi-OpenInventor-clients-2.1.5-10.i386.tar.gz`
- `tar xzvf sgi-OpenInventor-data-2.1.5-10.i386.tar.gz`
- `tar xzvf sgi-OpenInventor-devel-2.1.5-10.i386.tar.gz`

Probar el funcionamiento de Open Inventor

- `cd /usr/local/Inventor/share/data/models`
- `/usr/local/Inventor/bin/ivview modelo file.iv`

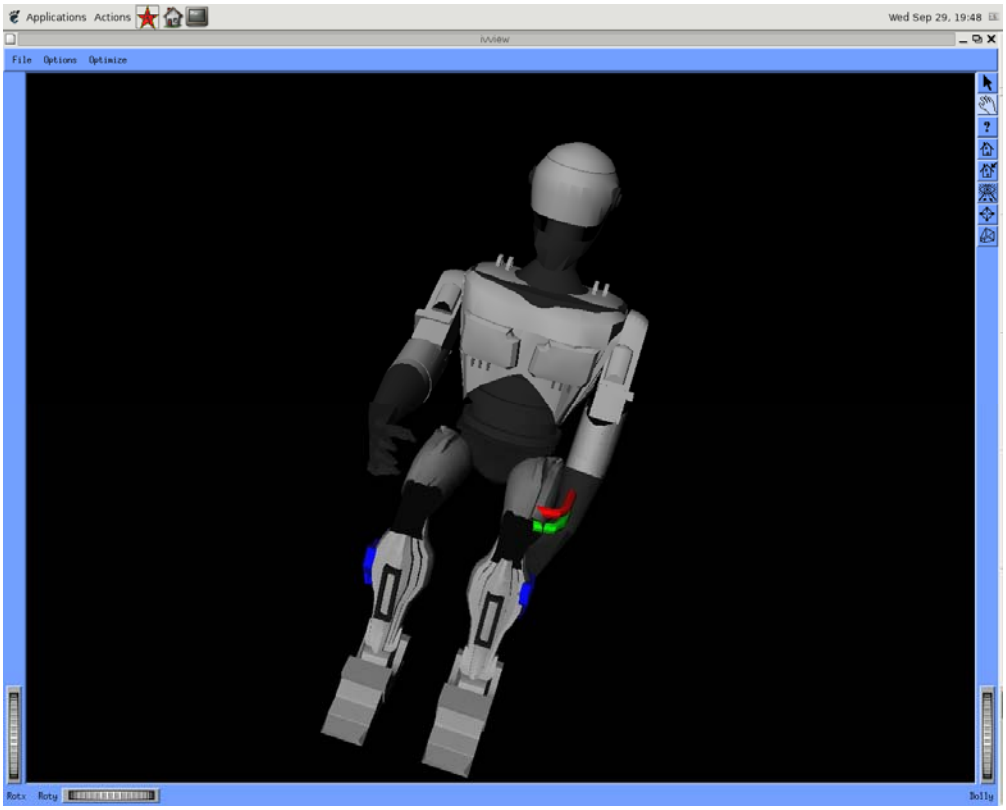

Figura 8.5 Ejemplo de OpenIvnetor

8.15 ITK

Es un conjunto de herramientas open source que son el sostén del proyecto Visible Human que pretende visualizar completamente en tres dimensiones el cuerpo humano. Funciona mejorando la segmentación y registro de datos. La segmentación es el proceso de identificar y clasificar los datos encontrados en la representación de una muestra digital. El registro de datos es la tarea de alinear o encontrar correspondencias entre datos. La segmentación y el registro de datos son aspectos fundamentales en cualquier aplicación medica y cuando son combinados adecuadamente con la visualización, son el soporte principal de aplicaciones prácticas [30].

Instalación:

- `Se descargan los Fuentes y se descomprimen.`
- `mkdir /usr/local/ITK-1.6.0/`
- `Mover los directorios de include y lib a la ruta /usr/local/ITK-1.6.0/`
- `Para ejecutar los ejemplos utilizar cmake`

Figura 8.6. Ejemplo de ITK

Conclusiones

En el presente trabajo se planteó como objetivo general la implementación de un cluster tipo Beowulf para aplicaciones de Visualización Científica y Realidad Virtual, documentando paso a paso diseño, instalación, configuración y administración de éste.

Para cumplir con este objetivo se trabajaron diversos aspectos, los cuales se enumeran a continuación.

En primer lugar, en cuanto a la elección de hardware a utilizar, se requirió de un análisis previo de las ventajas y desventajas de diferentes tecnologías de hardware. Dado que se trataba de un proyecto ya contemplado por el Departamento de Visualización, personal de dicha área ya se había dado a la tarea de hacer dicho análisis del hardware más óptimo para la construcción del cluster e incluso ya se había adquirido, por lo que la construcción del cluster Zion se llevaría a cabo utilizando este equipo.

Así pues, el análisis y elección de hardware a utilizar es una de las consideraciones más importantes, en este sentido se hace un poco difícil el adaptarse a cierta tecnología (hardware) existente, esto, debido a las características específicas del hardware (como las tarjetas gráficas, cantidad de memoria, espacio en disco, etc) lo que complicó y retraso un poco el proyecto. Sin embargo, de la experiencia adquirida, podemos concluir que los aspectos

más importantes a considerar en la elección del hardware para la construcción de un cluster están: la elección del procesador, el tipo de disco duro a utilizar (IDE o SCSI), la cantidad de memoria RAM, la tecnología de red a emplear y la elección de una tarjeta gráfica adecuada al tipo de aplicaciones a ejecutar.

Posteriormente, se debió hacer un análisis del sistema operativo a utilizar en el cluster; éste si fue llevado a cabo por los involucrados directos en el proyecto. Cabe aclarar que desde un principio se tenía claro que el cluster tendría que ser construido en su mayoría con herramientas de software de libre distribución. Para la elección del sistema operativo se hizo un análisis de los beneficios y bondades que ofrecen las diferentes distribuciones de Linux siendo el factor principal y determinante la disponibilidad del parche de RTAI para el kernel lo que redujo nuestro universo de posibilidades a distribuciones Linux dejando fuera aquellas *BSD ya que no hay soporte de RTAI para ello, decidiéndose finalmente por la distribución Debian Linux.

Una vez elegido el sistema operativo del cluster, y como recomendación y con ayuda del personal del Departamento de Visualización, se hizo un análisis de ciertas características especiales que requerían algunas aplicaciones a instalarse por lo que fue necesario la creación de un kernel personalizado para su correcta ejecución. En este sentido, un análisis previo permite tener un panorama bastante amplio sobre las características y requerimientos de instalación del cluster permitiendo una reducción de tiempos importante.

Para la administración de los recursos del cluster, se hizo necesario el diseño de un sistema de archivos acorde al tipo de aplicaciones a ejecutarse, también fue necesaria la creación de diferentes scripts escritos en Perl y Bash personalizados que permiten realizar tareas básicas de administración de usuarios, ejecución de comandos, seguridad y monitoreo del sistema.

Una vez instalado el cluster, se trabajó en la instalación y configuración de

las aplicaciones de visualización y realidad virtual ejecutadas en el cluster siendo un proceso que consumió gran cantidad de tiempo. Dado que el área de especialización de los participantes en la construcción del cluster no se relacionaba con aplicaciones gráficas o de realidad virtual, el personal del Departamento de Visualización se limitó a pedir la instalación de dichas aplicaciones y la ejecución de los ejemplos con que cuentan dichas aplicaciones. El proceso de pruebas por parte de personal especializado en el área de aplicación del cluster inició en el mes de noviembre de 2004 y está siendo llevado a cabo por personal del departamento y prestadores de servicio social.

En cuanto algunos puntos encontrados durante el proceso de instalación y administración del cluster podemos enumerar las siguientes conclusiones:

1. La planeación de un cluster requiere de la evaluación de diversos aspectos dependiendo del tipo de aplicaciones a ejecutar. Una buena planificación de nuestro cluster nos permite construirlo con firme objetivo de llevar al máximo su optimización y por consiguiente incrementar su desempeño apegándonos al presupuesto con que contamos.

2. En cuanto al hardware elegido, se cuenta con discos IDE, en este sentido hubiera sido preferible contar con discos SCSI por la tecnología empleada por estos últimos.

3. Se debería contar con algún esquema RAID o algún esquema de redundancia que permita al cluster contar con la característica de alta disponibilidad.

4. En el aspecto de la administración aunque ya existen una gran cantidad de herramientas implementadas para la administración de equipos de cómputo, algunas de ellas son demasiado complejas o sencillas para los requerimientos de un administrador por lo que la creación de scripts

personalizados que cumplan cabalmente las necesidades de un administrador son la mejor opción.

5. Una de las partes más importantes del proceso de administración de un cluster y de cualquier equipo de cómputo en general es la seguridad de éste. En este sentido, durante el proceso de instalación del nodo maestro del cluster y una vez que estuvo en red, hubieron varios intentos fallidos de intrusión, sin embargo, se tomó la decisión de reinstalar de nuevo el nodo e implementar medidas de seguridad básicas que evitaran la exposición de nuestro cluster. Así pues, una de las recomendaciones hechas y llevadas a cabo fué quitarlo de una red pública y permitir acceso local y remoto controlado .

6. Es necesario contar con un espacio físico adecuado con las condiciones ambientales óptimas para el óptimo desempeño del sistema ya que durante el proceso de construcción del cluster debido al calor generado por los equipos y a no contar con aire acondicionado, uno de los ventiladores de un nodo se descompuso por lo que fue necesario trabajar sin dicho nodo.

7. Este trabajo muestra que un cluster tipo Beowulf es una alternativa efectiva a una supercomputadora para la ejecución de programas y aplicaciones gráficas debido a su excelente relación costo/beneficio. Esto permite no sólo costos de instalación y operación más bajos sino incluso en algunos casos superar el rendimiento de supercomputadoras convencionales en determinados tipos de problemas aumentando el número de nodos del cluster.

8. Otra de las grandes ventajas de un cluster es que debido a su bajo costo, un grupo de trabajo con necesidades específicas o muy particulares puede disponer de los recursos computacionales suficientes.

Trabajo a futuro

Aunque en términos generales este trabajo cumplió con los objetivos inicialmente planteados, aún queda trabajo por hacer y consiste en:

- Instalación de nuevas aplicaciones.
- Adición de nuevos nodos al cluster.
- Configuración del sonido en el cluster.
- Configuración de nuevos dispositivos: Joystick, guantes,etc.
- La sincronización de proyectores que permita la generación de ambientes virtuales inmersivos.
- La implementación de un sistema de colas.
- Realización de métricas de rendimiento. Uno de los aspectos más importantes que aún hace falta por hacer, es un análisis a detalle del rendimiento del cluster tanto de su desempeño numérico como del gráfico, haciendo pruebas con benchmarks (pruebas de rendimiento) variando el número de procesadores y obteniendo estadísticas representativas.

Referencias

[1] Jorba, Josep. Suppi, Remo. *Administración Avanzada de GNU/Linux*. Barcelona, España. 2004.

[2] Intel: *Building High-Performance Computing Clusters with intel Architecture*. EUA. 2002

[3] Merkey, Phil. *Beowulf History*. 2004 http://www.beowulf.org/overview/history.html

[4] Alegría, Marcela. Sanabria, César. Vanegas, Verónica. *Cluster Optimizado para Visualización Científica y Realidad Virtual*. México. 2004

[5] Alegría, Marcela. Sanabria, César. Vanegas, Verónica. *Reporte de Implantación –configuración- administración de un cluster tipo beowulf para los departamentos de Visualización y Realidad Virtual*. México. 2004

[6] Cruz, Enrique. Gordillo José Luis. *Supercómputo en la UNAM*. México. 2004.

[7] Departamento de Supercómputo UNAM. *Clasificación de Arquitectura de Supercomputadoras*. México. 2003

[8] Bhamidipati, Prashant. Ganglia installation guide. 2003

[9] Blaise, Barney. *Introduction to Parallell Computing*. Livermore Computing. EUA. 2005

[10] García, Areli. (et-al). *Metodologías de Paralelización en la Supercomputadora. CICESE2000*. México. 2002. http://telematica.cicese.mx/computo/super/cicese2000/paralelo/

[11] Amy, Apon. Baker, Mark. *Tecnologías de Red*. Universidad de Arkansas, EUA. Univesidad de Prtosmouth, GB. 2002. http://clusters.unam.mx/Proyectos/white_paper/cap2.html

[12] The Mississippi Center for Supercomputing Research. *How to Build a Linux Cluster*. EUA. 2001. http://www.mcsr.olemiss.edu/bookshelf/articles/how_to_build_a_cluster.html

[13] Balderas, Roy. (et-al). *Administración del cluster tipo beowulf de la Facultad de Ingeniería*. Tesis de Licenciatura. Facultad de Ingeniería, UNAM. México. 2003

[14] Hernández, Liliana. *Diseño de un cluster beowulf computo de alto rendimiento para resolver problemas astrofisicos*. Tesis Maestria (Maestria en Ingenieria) UNAM. Mexico. 2004

[15] Cluster Zion, Home Page. http://www.depvis.unam.mx/cluster/

[16] vr++, Home Page. http://www.vrmedialab.dk/~vrpp/

[17] Performer, Home Page. http://www.sgi.com/software/performer/

[18] Vrjuggler, Home Page. http://www.vrjuggler.org/

[19] Netjuggler, Home Page. http://netjuggler.sourceforge.net/NetJuggler.php

[20] VTK, Home Page. http://public.kitware.com/VTK/

[21] Paraview, Home Page. http://www.paraview.org/

[22] Povray, Home Page. http://www.povray.org/

[23] OpenDX, Home Page. http://www.opendx.org/

[24] OpenScenegraph, Home Page. http://openscenegraph.sourceforge.net/

[25] vrNav, Home Page. http://www.ats.ucla.edu/portal/research_activities/vrNav/default.htm

[26] Chromium, Home Page. http://chromium.sourceforge.net/

[27] OpenSG, Home Page. http://www.opensg.org/

[28] Syzygy, Home Page. http://www.isl.uiuc.edu/ClusteredVR/ClusteredVR.htm

[29] OpenInventor, Home Page. http://oss.sgi.com/projects/inventor/

[30] ITK, Home Page. http://www.itk.org/

[31] RTAI, Home Page. http://www.rtai.org

[32] SoftGenLock, Home Page. http://netjuggler.sourceforge.net/SoftGenLock.php

[33] PVM, Home Page. http://www.epm.ornl.gov/pvm/pvm_home.html

[34] MPI, Home Page. http://www.lam-mpi.org/

[35] OpenGL, Home Page. http://www.opengl.org/

[36] Cluster de Computadoras. Conceptos Generales.
http://es.tldp.org/Manuales-LuCAS/doc-cluster-computadoras/doc-cluster-computadoras-html/node9.html

ANEXO A.

Documentación previa.

Dentro de este anexo se encuentran los documentos que definen el proyecto de construcción del cluster, sus características y requerimientos así como algunos más que se realizaron durante el proyecto.

Cluster Optimizado para Visualización Científica y Realidad Virtual

Becarios:	Carmen Marcela Alegría Cruz Verónica Vanegas Lomas César Sanabria Pineda
Responsables:	Luis Miguel de la Cruz Salas Ma. Carmen Ramos Nava
Área de especialización:	Administración
Área de aplicación:	Visualización Científica Realidad Virtual

1. Objetivo

El objetivo principal es la construcción de un arreglo de computadoras para plicaciones de visualización interactiva y distribuida de datos científicos en general, y la generación eficiente de ambientes virtuales por los que se pueda navegar interactivamente. En todas las aplicaciones se desea que las imágenes desplegadas por las diferentes componentes del arreglo aparezcan como una sola imagen, de tal manera que se deben sincronizar todas las componentes del arreglo mediante el uso de diferentes técnicas.

2. Herramientas

La infraestructura de hardware necesaria para la construcción del cluster, se encuentra en el Departamento de Visualización y consiste de lo siguiente:

Hardware:

4 nodos con:

2 Procesadores Athlon MP200 a 2Ghz
1Gb de RAM.
Tarjeta gráfica Nvidia Quadro4 900 GLX.
Disco de 80 Gb.
Tarjeta Fast Ethernet 1Gbit.

El software que se instalará en cada uno de los nodos del cluster para aplicaciones de Visualización Científica y Realidad Virtual es el siguiente:

Software:

Sistema operativo:

Linux Debian 2.6

Compiladores:

GNU 3.0.4 (c, c++, f77, java)

Software básico:

OpenGL
PVM
MPI
XMPI
PAPI
Lperfex

Visualización Científica:

OpenDX
VTK
ITK
POVRAY
Inventor (3 nodos)
Chromium

Realidad Virtual:

Performer
OpenSceneGraph
OpenSG
NetJuggler o VRjuggler 2.0
SoftGenLock
vrNav

3. Descripción

El cómputo paralelo y distribuido es actualmente una manera eficiente de resolver complejos problemas de ciencia e ingeniería. Los arreglos de computadoras personales (clusters) se han convertido en una opción viable y barata para resolver problemas de gran escala. Los clusters han probado su efectividad para resolver problemas clásicos de cómputo numérico. Recientemente, la introducción de tarjetas gráficas a bajo costo ha permitido el uso de los clusters en aplicaciones de visualización científica interactiva y de realidad virtual inmersiva, en las que se utiliza el despliegue a través de varios proyectores.

En el tipo de aplicaciones descrito en el párrafo anterior existe una dificultad mayor: sincronizar los recursos del cluster (CPU, memoria, GPU, proyectores) de tal manera que las imágenes desplegadas por los diferentes proyectores aparezcan como una sola imágen de alta resolución.

Las etapas que se consideran en el desarrollo de este proyecto son las siguientes:

1. Investigación de construcción de *clusters* para aplicaciones gráficas.
2. Conección de las diferentes partes del *cluster*.
3. Instalación y configuración del sistema operativo.
4. Instalación de software para las áreas de aplicación, para envío de mensajes y para medición de rendimiento.
5. Pruebas con modelos de varias áreas (no sincronización) y análisis de rendimiento.
6. Sincronización mediante software.
7. Pruebas con modelos de varias áreas y análisis de rendimiento.
8. Sincronización mediante hardware.
9. Pruebas con modelos de varias áreas y análisis de rendimiento.

Becarios

Carmen Marcela Alegría Cruz

Verónica Vanegas Lomas

Cesar Sanabria Pineda

Responsables

Luis Miguel de la Cruz Salas
Dept. de Visualización

Ma. Carmen Ramos Nava
Dept. de Realidad Virtual

3er coloquio estudiantil

PLAN DE BECARIOS EN CÓMPUTO DE ALTO RENDIMIENTO

MEMORIAS

GENERACIÓN 2003-2004

Carmen Marcela Alegria Cruz

Carlos Omar De La Rosa Delfín

Roberto David Esquivel Gutiérrez

Liliana Marcela González Suárez

Jesús Salim Hernández David

Cristina Orduña Rodríguez

Norma Araceli Reyes Alvarado

Elba Karén Sáenz García

Cesar Sanabria Pineda

Rafael Norman Saucedo Delgado

Omar Alejandro Solís Garza

Javier Torres Hernández

Oscar René Valdez Casillas

Verónica Vanegas Lomas

Alan Villegas Rendón

CLUSTER OPTIMIZADO PARA VISUALIZACIÓN CIENTÍFICA Y REALIDAD VIRTUAL

Becario	*Carmen Marcela Alegría Cruz*
	Cesar Sanabria Pineda
	Verónica Vanegas Lomas
Responsable del proyecto	*Ma. del Carmen Ramos Nava*
	Luis Miguel de la Cruz Salas
Área de especialización	*Administración de sistemas Unix*

Con la llegada de la primer supercomputadora (CRAY YMP/4) a la UNAM, la DGSCA ha proporcionado los recursos y mecanismos para procesar grandes volúmenes de información y dar solución a problemas de cómputo considerados importantes retos.

En la UNAM, a pesar de los recursos de supercómputo con los que se cuenta, es necesario crear y adaptar sistemas para el tratamiento de información con aplicaciones muy especificas, ofreciéndolos como una alternativa a áreas especializadas como visualización científica y realidad virtual.

Disponer de este tipo de infraestructura es por demás costoso, por ello, se considera a los clusters una de las opciones viables para atacar este tipo de problemas, ya que permiten obtener los recursos computacionales necesarios a un costo bastante aceptable.

El proyecto consistió en la construcción de un cluster para aplicaciones de visualización interactiva y distribuida de datos científicos, y la generación eficiente de ambientes virtuales inmersivos. Una de las características que debe cumplir este tipo de clusters es la sincronización de las imágenes que son generadas y desplegadas por los diferentes nodos del cluster en una sola imagen.

La sincronización de los componentes gráficos del cluster se realizó mediante la adición de características y módulos especiales al Kernel del sistema operativo.

El cluster se creó con una arquitectura tipo Beowulf, actualmente consiste de cuatro nodos duales (dos procesadores Athlon MP200 a 2GHz, 1GB de RAM por nodo, tarjeta gráfica Nvidia Quadro4 900GLX), bajo el sistema operativo Debian Linux y conectados mediante una red gigabit Ethernet.

Las aplicaciones básicas instaladas son: Real Time Application Interface (RTAI), SoftGenLock, PVM, MPI, OpenGL. El software de aplicaciones gráficas y de realidad virtual disponibles en el cluster son: vr++, OpenDX, VTK, ITK, POVRAY, OpenInventor, Performer, OpenSceneGraph, OpenSG, NetJuggler, VRJuggler, vrNav, Syzygy.

ANEXO B.

Scripts y Archivos de Configuración

En este anexo se encuentran los archivos de configuración y scritps de administración, seguridad y monitoreo más importantes del cluster.

Archivo de configuración /etc/X11/XF86Config-4

```
zion0:~# cat /etc/X11/XF86Config-4
### BEGIN DEBCONF SECTION
# XF86Config-4 (XFree86 server configuration file) generated by dexconf, the
# Debian X Configuration tool, using values from the debconf database.
#
# Edit this file with caution, and see the XF86Config-4 manual page.
# (Type "man XF86Config-4" at the shell prompt.)
#
# If you want your changes to this file preserved by dexconf, only make changes
# before the "### BEGIN DEBCONF SECTION" line above, and/or after the
# "### END DEBCONF SECTION" line below.
#
# To change things within the debconf section, run the command:
#   dpkg-reconfigure xserver-xfree86
# as root.  Also see "How do I add custom sections to a dexconf-generated
# XF86Config or XF86Config-4 file?" in /usr/share/doc/xfree86-common/FAQ.gz.

Section "Files"
        FontPath        "unix/:7100"                    # local font server
        # if the local font server has problems, we can fall back on these
        FontPath        "/usr/lib/X11/fonts/Type1"
        FontPath        "/usr/lib/X11/fonts/CID"
        FontPath        "/usr/lib/X11/fonts/Speedo"
        FontPath        "/usr/lib/X11/fonts/misc"
        FontPath        "/usr/lib/X11/fonts/cyrillic"
        FontPath        "/usr/lib/X11/fonts/100dpi"
        FontPath        "/usr/lib/X11/fonts/75dpi"
EndSection

Section "Module"
        Load    "bitmap"
        Load    "dbe"
        Load    "ddc"
        Load    "extmod"
        Load    "freetype"
        Load    "glx"
        Load    "int10"
        Load    "record"
```

```
        Load    "speedo"
        Load    "type1"
        Load    "vbe"
EndSection

Section "InputDevice"
        Identifier      "Generic Keyboard"
        Driver          "keyboard"
        Option          "CoreKeyboard"
        Option          "XkbRules"      "xfree86"
        Option          "XkbModel"      "pc105"
        Option          "XkbLayout"     "us"
EndSection

Section "InputDevice"
        Identifier      "Configured Mouse"
        Driver          "mouse"
        Option          "CorePointer"
        Option          "Device"                "/dev/psaux"
        Option          "Protocol"              "ImPS/2"
        Option          "Emulate3Buttons"       "true"
        Option          "ZAxisMapping"          "4 5"
EndSection

Section "InputDevice"
        Identifier      "Generic Mouse"
        Driver          "mouse"
        Option          "SendCoreEvents"        "true"
        Option          "Device"                "/dev/input/mice"
        Option          "Protocol"              "ImPS/2"
        Option          "Emulate3Buttons"       "true"
        Option          "ZAxisMapping"          "4 5"
EndSection

Section "Device"
        Identifier      "NVIDIA"
        Driver          "nvidia"
        VideoRam        131072

        Option      "TwinView" "true"
        Option      "TwinViewOrientation" "Clone"
        Option      "SecondMonitorHorizSync" "30.0 - 121.0"
        Option      "SecondMonitorVertRefresh" "48.0 - 160.0"
        Option      "Stereo" "3"

EndSection

Section "Monitor"
        Identifier      "Monitor"
        HorizSync       30-121
```

```
        VertRefresh     48-160
        Option          "DPMS"
EndSection

Section "Screen"
        Identifier      "Default Screen"
        Device          "NVIDIA"
        Monitor         "Monitor"
        DefaultDepth    24
        SubSection "Display"
                Depth           1
                Modes           "1280x1024" "1280x960" "1152x864" "1024x768"
"800x600" "640x480"
        EndSubSection
        SubSection "Display"
                Depth           4
                Modes           "1280x1024" "1280x960" "1152x864" "1024x768"
"800x600" "640x480"
        EndSubSection
        SubSection "Display"
                Depth           8
                Modes           "1280x1024" "1280x960" "1152x864" "1024x768"
"800x600" "640x480"
        EndSubSection
        SubSection "Display"
                Depth           15
                Modes           "1280x1024" "1280x960" "1152x864" "1024x768"
"800x600" "640x480"
        EndSubSection
        SubSection "Display"
                Depth           16
                Modes           "1280x1024" "1280x960" "1152x864" "1024x768"
"800x600" "640x480"
        EndSubSection
        SubSection "Display"
                Depth           24
                Modes           "1280x1024" "1280x960" "1152x864" "1024x768"
"800x600" "640x480"
        EndSubSection
EndSection

Section "ServerLayout"
        Identifier      "Default Layout"
        Screen          "Default Screen"
        InputDevice     "Generic Keyboard"
        InputDevice     "Configured Mouse"
        InputDevice     "Generic Mouse"
EndSection

Section "DRI"
```

```
        Mode    0666
EndSection

### END DEBCONF SECTION
```

Script alta_usuario

```
zion0:/usr/local/sbin/admon# cat alta_usuario
#!/usr/bin/perl
# Alta de usuarios
# 26 julio 2004
# -------------------------------------

 if (($#ARGV) == -1 ){
  print " \n Use: $0 login \n";
  exit;
 }
 else{
  chomp($USER=`whoami`);
  chomp($HOST=`hostname`);
  if ($USER =~ 'root'){
   if ( $HOST =~ 'zion2' ){
    $USER_ADICIONAR=$ARGV[0];
    chomp($USER_ADICIONAR);
    open FH, "</etc/passwd" or die "cannot open /etc/passwd";
    while ( <FH> ) {
     chomp;
     ($login,$extra)= split(/:/,$_,2);
     $total{$login}{extra}=$extra;
    }
    close FH;
    if ((exists($total{$USER_ADICIONAR}))){
     print "\n $USER_ADICIONAR ya es un usuario valido \n";
     exit;
    }
    print "Validando ...";
    $x=`/usr/local/sbin/admon/host-up.pl`;
    if ( -f "/var/tmp/OK" ){
     print "...OK\n";
     print(`cat /var/tmp/host-up-msg`);
     $x=`rm -rf /var/tmp/OK`;
     do{
      print "\n El nuevo usuario pertenecera a :\n";
      print " 1 Visualizacion \n";
      print " 2 RV \n\n";
      print " Opcion? ";
      chomp($res=<STDIN>);
     }until($res==1 | $res==2 );
     if ($res == 1){
      if (($pid=fork()) == 0 ){
       $ENV{USER_ADD} = $USER_ADICIONAR;
      # @args = ("/usr/sbin/useradd", "-g","1006","$ENV{USER_ADD}");
       @args = ("/bin/echo", "-g","1006","$ENV{USER_ADD}");
       system(@args) == 0 or die "system @args failed: $?";
```

```
       exit;
      }
      wait;
      exit;
     }
     else {
      if (($pid=fork()) == 0 ){
       $ENV{USER_ADD} = $USER_ADICIONAR;
       @args = ("/usr/sbin/useradd", "-g","1005","$ENV{USER_ADD}");
       system(@args) == 0 or die "system @args failed: $?";
       exit;
      }
      wait;
     }

    # Sincronizar  passwd & group en todos los nodos
     $x=`/usr/bin/rsync /etc/passwd zion0:/etc/passwd`;
     $x=`/usr/bin/rsync /etc/group zion0:/etc/group`;
     $x=`/usr/bin/rsync /etc/shadow zion0:/etc/shadow`;

    # Genera llaves
    print "**************************************************\n";
    print "*   Recuerda loogerate con el usuario nuevo     *\n";
    print "* y crearle sus llaves por medio de intercambia *\n";
    print "**************************************************\n";
   }
   else {
    print(`cat /var/tmp/host-up-msg`);
    print "Fallo!! \n";
    print "\n Probablemente la causa: algun nodo no esta arriba o esta en
win2\n";
    exit 1;
   }
  }
  else{
   print "**************************************************\n";
   print "*    $USER:  Recuerda que debes estar en zion2   *\n";
   print "* para poder dar de alta usuarios               *\n";
   print "**************************************************\n";
   exit 0;
  }
 }
 else{
  print "**************************************************\n";
  print "    $USER:  tu no puedes crear usuarios      \n";
  print "**************************************************\n";
  exit 0;
 }
 }
```

Como puede verse en el código anterior, el script llama a un archivo llamado host-up.pl que en términos generales lo que hace es verificar que todos y cada uno de los nodos que componen el cluster se encuentren funcionando y ejecutando linux dado que el nodo 1 cuenta con dos sistemas operativos. El código del script es:

```
zion0:/usr/local/sbin/admon# cat host-up.pl
#!/usr/bin/perl
# 19/agosto/2004
# Valida que los nodos este arriba y en Linux
#==================================================

 `rm -f /var/tmp/host-up-msg`;
 chomp($h0=`nmap -sP zion0 | grep Host| head -1 `);
 if ($h0 =~ 'Host zion0 \(192.168.0.2\) appears to be up.'){
   print("zion0 arriba\n");
   open(FH,">>/var/tmp/host-up-msg");
   print FH "Zion0 arriba [ OK ]\n";
   chomp($h1=`nmap -sP zion1 | grep Host | head -1`);
   chomp($h11=`nmap -O zion1 | grep \"OS details\"| head -1 `);
   if ($h1 =~ 'Host zion1 \(192.168.0.3\) appears to be up.'){
     print("zion1 arriba\n");
     print FH "Zion1 arriba [ OK ]\n";
     if ($h11 =~ 'OS details: Microsoft Windows Millennium Edition \(Me\),
Windows 2000 Professional or Advanced Server, or Windows XP'){
        print FH "Zion1 esta en Windows, para a~adir usuario debe correr
Linux\n";
        close FH;
     }
     else{
      chomp($h2=`nmap -sP zion2 | grep Host |head -1`  );
      if ($h2 =~ 'Host zion2 \(192.168.0.4\) appears to be up.'){
        print("zion2 arriba\n");
        print FH "Zion2 arriba [ OK ]\n";
        chomp($h3=`nmap -sP zion3 | grep Host |head -1 `);
        if ($h3 =~ 'Host zion3 \(192.168.0.5\) appears to be up.'){
          print("zion3 arriba\n");
          print FH "Zion3 arriba [ OK ]\n";
          print FH "Cluster nodes [ OK ]\n";
          close FH;
          print("Cluster nodes [ OK ]\n");
          `touch /var/tmp/OK`;
        }
###Todo el codigo restante
        elsif($h3 =~ 'Note: Host seems down. If it is really up, but blocking
our ping probes, try -P0'){
          print FH "No se puede a~adir usuario, el nodo zion3 no responde\n";
```

```
          close FH;
        }
       }
       elsif($h2 =~ 'Note: Host seems down. If it is really up, but blocking our
ping probes, try -P0'){
         print FH "No se puede a~adir usuario, el nodo zion2 no responde\n";
         close FH;
       }
      }
     }
     elsif($h1 =~ 'Note: Host seems down. If it is really up, but blocking our
ping probes, try -P0'){
       print FH "No se puede a~adir usuario, el nodo zion1 no responde\n";
       close FH;
     }
    }
    elsif($h0 =~ 'Note: Host seems down. If it is really up, but blocking our
ping probes, try -P0'){
      print FH "No se puede a~adir usuario, el nodo zion0 no responde\n";
      close FH;
    }
   else{
     print("***** No esta validando\n");
   }
```

Script baja_usuario

```
zion0:/usr/local/sbin/admon# cat baja_usuario
#!/usr/bin/perl
# baja de usuarios
# 27 septiembre 2004
# -------------------------------------

 chomp($SOY=`whoami`);
 chomp($HOST=`hostname`);
 if ($SOY =~ 'root'){
  if ( $HOST =~ 'zion2' ){
   if (($#ARGV+1) == 1 ){
    print "Validando ...";
    $x=`/usr/local/sbin/admon/host-up.pl`;
    if ( -f "/var/tmp/OK" ){
     print "...OK\n";
     print(`cat /var/tmp/host-up-msg`);
     $x=`rm -rf /var/tmp/OK`;
     do{
      print "\nMenu: \n";
      print "\n1. Desactivar cuenta ";
      print "\n2. Eliminar cuenta ";
      print "\n3. Cancelar accion ";
      print "\n\n   Opcion: ";
      chomp($opcion=<STDIN>);
     }until($opcion>=1 && $opcion<=3);
     if ( $opcion == 3){
      print " Accion cancelada... no hubo ningun cambio \n";
      exit;
     }
     else{
      $USER_ELIMINAR=$ARGV[0];
      chomp($USER_ELIMINAR);
      open FH, "</etc/passwd" or die "cannot open /etc/passwd";
      while ( <FH> ) {
       chomp;
       ($login,$extra)= split(/:/,$_,2);
       $total{$login}{extra}=$extra;
      }
      close FH;
      if (!(exists($total{$USER_ELIMINAR}))){
       print "\n $USER_ELIMINAR no es un usuario valido \n";
       exit;
      }
      else{
       if ( $opcion == 1 ){
        do{
         print "\n\n Esta seguro de deshabilitar a ".$USER_ELIMINAR." [Y/N]: ";
```

```
    chomp($res=<STDIN>);
    $res=uc($res);
   }until ( $res =~ /[YN]/ );
   if ($res =~ /Y/ ){
    if (($pid=fork()) == 0 ){
     $ENV{USER_DESHABILITAR} = $USER_ELIMINAR;
     @args = ("/usr/sbin/usermod", "-L","$ENV{USER_DESHABILITAR}");
     system(@args) == 0 or die "system @args failed: $?";
     exit;
    }
    wait;
    # Sincronizar shadow en todos los nodos
    print "\n Sincronizando shadow ...";
    $x=`/usr/bin/rsync /etc/shadow zion0:/etc/shadow`
    print "... [ OK ]\n";
   }
   else{
    print "\n\n NOTA:  $USER_ELIMINAR no fue desactivado\n";
    exit;
   }
  }
  else{
   do{
    print "\n\n Eliminar a $USER_ELIMINAR y todos sus archivos [Y/N]? ";
    chomp($res=<STDIN>);
    $res=uc($res);
   }until ( $res =~ /[YN]/ );
   if ($res =~ /Y/ ){
    if (($pid=fork()) == 0 ){
     $ENV{USER_ELIMINAR} = $USER_ELIMINAR;
     @args = ("/usr/sbin/deluser", "--remove-home","$ENV{USER_ELIMINAR}");
     system(@args) == 0 or die "system @args failed: $?";
     exit;
    }
    wait;
    # Sincronizar passwd & group en todos los nodos
    print "\n Sincronizando passwd & group ...";
    $x=`/usr/bin/rsync /etc/passwd zion0:/etc/passwd`
    $x=`/usr/bin/rsync /etc/group zion0:/etc/group`
    print "... [ OK ]\n";
   }
   else{
    print "\n\n NOTA:  $USER_ELIMINAR no fue eliminado\n";
    exit;
   }
  }
 }
}

#  Sincronizar shadow en todos los nodos
```

```
    print "\n Sincronizando shadow ...";
    $x=`/usr/bin/rsync /etc/shadow zion0:/etc/shadow`
    print "... [ OK ]\n";

    # mensaje a bitacoras o bien un mail a root
    print "\n Debes mandar mensaje a bitacoras?\n";
    # mail root -s "usuario " < /tmp/error.log
   }
   else{
    $z=`cat /tmp/host-up-msg`;
    print("$z");
    print "Fallo!! \n";
    print "\n Probablemente algun nodo no esta arriba o esta en win2\n";
    exit;
   }
  }
  else{
   print "\n Use:  $0 login \n";
   exit;
  }
 }
 else{
  print "*************************************************\n";
  print "    $SOY:  Recuerda que debes estar en zion2   \n";
  print " para poder dar de baja a usuarios             \n";
  print "*************************************************\n\n";
  exit 0
 }
}
else{
 print "*****************************************************\n";
 print " $SOY:  tu no puedes deshabiltar/eliminar usuarios  \n";
 print "*****************************************************\n\n";
 exit 0
}
```

Archivos de configuración de TCP_Wrappers

```
zion0:~# cat /etc/hosts.deny
# /etc/hosts.deny: list of hosts that are _not_ allowed to access the system.
#                  See the manual pages hosts_access(5), hosts_options(5)
#                  and /usr/doc/netbase/portmapper.txt.gz
#
# Example:    ALL: some.host.name, .some.domain
#             ALL EXCEPT in.fingerd: other.host.name, .other.domain
#
# If you're going to protect the portmapper use the name "portmap" for the
# daemon name. Remember that you can only use the keyword "ALL" and IP
# addresses (NOT host or domain names) for the portmapper. See portmap(8)
# and /usr/doc/portmap/portmapper.txt.gz for further information.
#
# The PARANOID wildcard matches any host whose name does not match its
# address. You may wish to enable this to ensure any programs that don't
# validate looked up hostnames still leave understandable logs. In past
# versions of Debian this has been the default.
#ALL: PARANOID
#ALL:ALL
ALL: ALL: spawn ( \
echo -e "\n\
TCP Wrappers\: Connection Refused\n\
By\:           $(uname -n)\n\
Process\:      %d (pid %p)\n\
Host\:         %c\n\
Date\:         $(date)\n\
From tcpd@$(uname -n). %u@%h -> %d.\n\
%h\n\
" | /usr/local/sbin/admon/alert.pl )
```

```
zion0:~# cat /etc/hosts.allow
# /etc/hosts.allow: list of hosts that are allowed to access the system.
#                   See the manual pages hosts_access(5), hosts_options(5)
#                   and /usr/doc/netbase/portmapper.txt.gz
#
# Example:    ALL: LOCAL @some_netgroup
#             ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
#
# If you're going to protect the portmapper use the name "portmap" for the
# daemon name. Remember that you can only use the keyword "ALL" and IP
```

```
# addresses (NOT host or domain names) for the portmapper. See portmap(8)
# and /usr/doc/portmap/portmapper.txt.gz for further information.
#
# servicio al exterior
# 132.248.124.116 - 192.168.0.1
sshd:ALL

# Servicio a nodos
# sera usado para nfs
ALL:192.168.0.0/255.255.255.0
```

FIREWALL

```
#!/bin/bash
# script de inicilizacion del firewall (Cesar Sanabria Pineda 30-04-2004)
##############################################################################
#################
#
#
# UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
#
# DIRECCION GENERAL DE COMPUTO ACADEMICO
# FIREWALL DE SEGURIDAD    #
# CLUSTER PARA VISUALIZACION CIENTIFICA
##############################################################################
#################


        #########################
        # Definicion de variables
        #########################

        IPTABLES="/sbin/iptables"
        LAN="192.168.0.0/8"

        #
        # Informacion sobre este host
        #
        INET_IP="132.248.124.116"
        INET_IFACE="eth0"
        LO_IP="127.0.0.1"
        LO_IFACE="lo"

        ##################################################################
        #Configuracion POLITICAS iniciales por default.
        #Estas reglas aplican si las demas reglas no lo hacen
        #Por default todos los paquetes enviados hacia el firewall o la red
        #interna se tiran y, se permite todo el trafico de salida
        ################################################################
        set_policies()
            {
            $IPTABLES -P INPUT DROP
            $IPTABLES -P FORWARD DROP
            $IPTABLES -P OUTPUT ACCEPT
    }

##############################################
#LIMPIAR (-F) TODAS LAS REGLAS DE LA TABLA
#DE LA TABLA DE FILTRADO
##############################################
clean_rules()
```

```
    {
    $IPTABLES -F INPUT
    $IPTABLES -F FORWARD
    $IPTABLES -F OUTPUT
    $IPTABLES -F -t nat
    }

##############################################
#PERMITE CONEXIONES ESTABLECIDAS Y RELACIONADAS
#EN TRAFICO DE ENTRADA Y SALIDA
##############################################
allow_established_conn()
    {
    $IPTABLES -A FORWARD -i eth0 -o eth1 -m state --state ESTABLISHED,RELATED -j
ACCEPT
    #Permitir  paquetes  de  entrada  en  el  mismo  firewall  que  son  parte  de
conexiones
    #existentes relacionadas.
    $IPTABLES -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
    $IPTABLES -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
    }


#El resto contiene reglas especificas que se aplican en orden como se
#listan. Si ninguna aplica, entonces las reglas anteriores se ejecutan.

#################################################################
#PERMITIR PAQUETES DE eth1 A eth0 (RED INTERNA -> INTERNET).
#################################################################
allow_internal_traffic()
   {
   $IPTABLES -A FORWARD -m state --state NEW -i eth1 -o eth0 -j ACCEPT
   }

allow_internal_connections()
   {
   #Permitir todas las entradas hacia el firewall desde la red interna e
   #interfaces locales
   $IPTABLES -A INPUT -i eth1 -s 192.168.0.0/24 -d 0/0 -j ACCEPT
   $IPTABLES -A INPUT -i lo -s 192.168.0.0/24 -d 0/0 -j ACCEPT
   }

allow_snat_eth0()
   {
   #Habilitar la fucionalidad SNAT sobre eth0.
   #SNAT es usado para traducir IP's privadas de interfaces dentro de la LAN
   #hacia la IP homologada.
   $IPTABLES -A POSTROUTING -t nat -s 192.168.0.0/24 -o eth0 -j SNAT --to-source
132.248.124.116
   }
```

```
drop_outbound_traffic()
   {
   #Negar todos los paquetes que provenga de internet hacia la interfaz eth0
   #los cuales han querido penetrar en direcciones homologadas en la LAN
   #
   $IPTABLES -A INPUT -i eth0 -s 132.248.124.116/32 -j DROP
   $IPTABLES -A INPUT -i eth0 -s 192.168.0.0/24 -j DROP
   $IPTABLES -A INPUT -i eth0 -s 127.0.0.0/8 -j DROP
   }

allow_servers()
   {
   # Servicios proporcionados por el cluster al exterior
   # Limitar el acceso al puerto 80 a IP 132.248.124.101

   $IPTABLES -A INPUT -i eth0 -s 132.248.124.101 -p tcp -m state --state NEW -m
tcp -d 132.248.124.116 --dport 80 -j ACCEPT

# Acceso a conexiones SSH

   $IPTABLES -A INPUT -i eth0 -s 132.248.124.101 -p tcp -m state --state NEW -m
tcp -d 132.248.124.116 --dport 22 -j ACCEPT # Marce, Vero y Cesar
# Lista de ip's permitidas
   }

allow_dns()
   {
   #Autorizar al servidor DNS, para dinstinguir la peticion de la red publica.
   $IPTABLES -A INPUT -p udp -m state --state NEW -m udp -d 132.248.10.2 --dport
53 -j ACCEPT
   }

drop_udp()
   {
   #Finalmente negar todas las peticiones de puertos UDP restantes para todas
las
   #conexiones SYN hacia cualquier puerto TCP no especificado en reglas
   #anteriores.
   $IPTABLES -A INPUT -s 0/0 -d 0/0 -p udp -j DROP
   $IPTABLES -A INPUT -s 0/0 -d 0/0 -p tcp --syn -j DROP
   }


#################
#TRAFICO LOCAL
#################
allow_local_traffic()
    {
    $IPTABLES -A INPUT -s 127.0.0.1 -i lo -j ACCEPT
```

```
    $IPTABLES -A INPUT -s 132.248.124.116 -i lo -j ACCEPT
    $IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LO_IP -j ACCEPT
    $IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $INET_IP -j ACCEPT
    }


##############################
allow_outbound_ping_traffic()
    # Va a permitir ping
    {
    $IPTABLES -A OUTPUT -d $LAN -p icmp -m state --state NEW -m icmp --icmp-type
8 -j ACCEPT
    }

##############################
# MAIN
##############################
start()
{
        echo -n $"Starting firewall:"
        echo
        # Permitir el envio de paquetes IP desde red interna,
        #habilitando IP forwarding
                #
        echo 1 > /proc/sys/net/ipv4/ip_forward
        #
        #Prevencion de SYN floods para el consumo de recursos en memoria:
        #
        #echo 1 > /proc/sys/net/ipv4/tcp_syncookies
        #
        set_policies
        clean_rules
        allow_local_traffic
        allow_established_conn
        allow_internal_traffic
        allow_internal_connections
        allow_local_traffic
        allow_servers
        allow_snat_eth0
        allow_dns
        #Para serivicio ntpd
        $IPTABLES -A INPUT -p tcp -m state --state NEW -m tcp --dport 123 -j
ACCEPT
        $IPTABLES -A INPUT -p udp -m state --state NEW -m udp --dport 123 -j
ACCEPT
        #icmp_protocol
        $IPTABLES -A INPUT -p icmp -m state --state NEW -m icmp --icmp-type 8 -j
ACCEPT
        $IPTABLES -A OUTPUT -p tcp -m state --state NEW -m tcp  --dport 22 -j
ACCEPT
```

```
        $IPTABLES -A OUTPUT -p icmp -m state --state NEW -m icmp --icmp-type 8 -
j ACCEPT
        $IPTABLES -A INPUT -p udp -j DROP
        allow_outbound_ping_traffic
        drop_outbound_traffic
        #Estara mandandoa logs todo aquello que no sea trafico permitido
        #$IPTABLES -A INPUT -j LOG --log-prefix "Bad-input-zion0:"
        #$IPTABLES -A OUTPUT -j LOG --log-prefix "Bad-output-zion0:"
        RETVAL=$?
        [ "$RETVAL" = 0 ] && touch /var/lock/subsys/firewall
        echo -n $"Ok! Firewall started"
        echo
}

stop()
{
        echo -n $"Stopping firewall (you really shouldn't ...):"
        echo
        $IPTABLES -P INPUT ACCEPT
        $IPTABLES -P FORWARD DROP
        $IPTABLES -P OUTPUT ACCEPT
        $IPTABLES -F INPUT
        $IPTABLES -F FORWARD
        $IPTABLES -F OUTPUT
        RETVAL=$?
        [ "$RETVAL" = 0 ] && rm -f /var/lock/subsys/sshd
        echo -n $"Firewall stopped"
        echo
}

case "$1" in
        start)
                start
                ;;
        stop)
                stop
                ;;
        restart)
                stop
                start
                ;;
        reload)
                reload
                ;;
        *)
                echo $"Usage: $0 {start|stop|restart}"
                RETVAL=1
esac
exit $RETVAL
```

Archivo de configuración /etc/sudoers

```
zion0:~# cat /etc/sudoers
# sudoers file.
# This file MUST be edited with the 'visudo' command as root.
# See the sudoers man page for the details on how to write a sudoers file.
Defaults logfile=/var/log/sudo.log
#-------------------------------------------------------------------------------
--
# Host alias specification
# Hosts en los que se debe tener permisos de ejecutar
#-------------------------------------------------------------------------------
--
Host_Alias           NODOS=192.168.0.1,  192.168.0.2,  192.168.0.3,  192.168.0.4,
192.168.0.5
Host_Alias      HOME=192.168.0.3
#-------------------------------------------------------------------------------
--
# User alias specification
#-------------------------------------------------------------------------------
--
# --- Usuarios ADMINISTRADORES
User_Alias      ADMIN=admin, marce, cesar, vero, admin
User_Alias      USERS=cesar, marce
#-------------------------------------------------------------------------------
--
# Cmnd alias specification
#-------------------------------------------------------------------------------
--
Cmnd_Alias      GOD=/bin/su
Cmnd_Alias              INSTALL=/usr/bin/apt-get,/usr/bin/dpkg,  /usr/sbin/dpkg-
reconfigure, /usr/bin/apt-show
Cmnd_Alias                     USERADMIN=/usr/sbin/useradd,   /usr/sbin/userdel,
/usr/sbin/usermod
Cmnd_Alias      USUARIOS=/usr/local/sbin/admon/envia
Cmnd_Alias      GROUPADMIN=/usr/sbin/grpadd, /usr/sbin/grpdel, /usr/sbin/grpmod
Cmnd_Alias      FILESADMIN=/bin/chmod, /bin/chown, /bin/chgrp
Cmnd_Alias      SEGURIDAD=/sbin/iptables


#-------------------------------------------------------------------------------
--
```

```
# User privilege specification
#-------------------------------------------------------------------------------
--
# --- El superusuario tiene todos los permisos ---
root    ALL=(ALL) ALL
# --- Para los Nodos en produccion
ADMIN           NODOS=GOD, USERADMIN, GROUPADMIN, FILESADMIN, SEGURIDAD
USERS           NODOS=NOPASSWD: USUARIOS        #Para cambio de passwords
```

Ejemplo de correo enviado por LOGWATCH

```
From root@zion0.depvis.unam.mx Wed Mar 30 06:25:19 2005
Return-path: <root@zion0.depvis.unam.mx>
Envelope-to: root@zion0.depvis.unam.mx
Delivery-date: Wed, 30 Mar 2005 06:25:19 -0600
Received: from root by zion0 with local (Exim 4.44)
        id 1DGcG7-0008HU-Ca
        for root@zion0.depvis.unam.mx; Wed, 30 Mar 2005 06:25:19 -0600
To: root@zion0.depvis.unam.mx
Subject: LogWatch for zion0
Message-Id: <E1DGcG7-0008HU-Ca@zion0>
From: root <root@zion0.depvis.unam.mx>
Date: Wed, 30 Mar 2005 06:25:19 -0600
Status: R
 ################### LogWatch 5.2.2 (06/23/04) ####################
       Processing Initiated: Wed Mar 30 06:25:01 2005
       Date Range Processed: yesterday
     Detail Level of Output: 5
          Logfiles for Host: zion0
 ################################################################
 --------------------- Cron Begin ------------------------
Commands Run:
   User root:
         rm -f /var/log/autologin ; killall -9 rungetty >/dev/null 2>&1: 288
Time(s)
         run-parts --report /etc/cron.hourly: 24 Time(s)
        [ -d /var/lib/php4 ] && find /var/lib/php4/ -type f -cmin
+$(/usr/lib/php4/maxlifetime) -print0 | xargs -r -0 rm: 48
Time(s)
      /usr/local/sbin/admon/ssync: 1 Time(s)
      test -e /usr/sbin/anacron || run-parts --report /etc/cron.daily: 1
Time(s)
 ---------------------- Cron End -------------------------

 --------------------- httpd Begin ------------------------
0.00 MB transfered in 6038 responses  (1xx 0, 2xx 5948, 3xx 44, 4xx 46, 5xx 0)
 1344 Images (0.00 MB),
 4690 Content pages (0.00 MB),
 4 Redirects (0.00 MB),
 ---------------------- httpd End -------------------------

 --------------------- pam_unix Begin ------------------------
cron:
   Sessions Opened:
```

```
      root: 362 Time(s)

gdm:
   Sessions Opened:
      nara: 1 Time(s)
   Unknown Entries:
      authentication  failure;  logname=  uid=0  euid=0  tty=:0  ruser=  rhost=
user=nara: 1 Time(s)
sshd:
   Authentication Failures:
      cesar (akbal.depvis.unam.mx): 1 Time(s)
   Sessions Opened:
      root: 9 Time(s)
      cesar: 5 Time(s)
      marce: 2 Time(s)

su:
   Sessions Opened:
      cesar(uid=0) -> root: 5 Time(s)
      (uid=0) -> nobody: 1 Time(s)
      cesar(uid=0) -> marce: 1 Time(s)
      marce(uid=0) -> root: 1 Time(s)

sudo:
   Unknown Entries:
      authentication  failure;  logname=cesar  uid=0  euid=0  tty=pts/0  ruser=
rhost=  user=cesar: 1 Time(s)
 ---------------------- pam_unix End -------------------------

 --------------------- SSHD Begin ------------------------
Users logging in through sshd:
   cesar:
      akbal.depvis.unam.mx (132.248.124.227): 5 times
   marce:
      tecelic.depvis.unam.mx (132.248.124.211): 2 times
   root:
      zion0 (192.168.0.2): 9 times

Error in PAM authentication:
   Authentication failure for cesar from akbal.depvis.unam.mx : 1 Time(s)
 ---------------------- SSHD End -------------------------

 --------------------- Sudo (secure-log) Begin ------------------------
==============================================================================
```

```
cesar => root
--------------------------------------------------------------------------------
/bin/su -
/bin/su -
/bin/su -
/bin/su -
/bin/su -
================================================================================
marce => root
--------------------------------------------------------------------------------
/bin/su -
Unmatched Entries:
(pam_unix) authentication failure; logname=cesar uid=0 euid=0 tty=pts/0 ruser=
rhost=  user=cesar
 ---------------------- Sudo (secure-log) End -------------------------

 --------------------- Syslogd Begin ------------------------
Syslogd started 1 Time(s)
 ---------------------- Syslogd End -------------------------

------------------ Disk Space --------------------
/dev/hda1             4.6G  3.4G 1007M  78% /
/dev/hda2              46G   38M   44G   1% /tmp
/dev/hda3              62G  5.2G   54G   9% /usr
 ###################### LogWatch End #########################
```

ANEXO C.

Algunas métricas de rendimiento.

Rendimiento

Una de las parte importantes en un cluster es la medición de las diferentes pruebas de su rendimiento. En este sentido, se realizaron un par de benchmarks (pruebas de rendimiento), uno para medición de su desempeño al realizar cálculos y otro para las métricas de su desempeño gráfico.

El benchmark para desempeño de cálculos utilizado fué Linpack que es un conjunto de bibliotecas matemáticas en Fortran para la solución de sistemas de ecuaciones lineales.

Así pues los resultados obtenidos son los siguientes:
Linux Debian Kernel 2.4.24
Mpich 1.2.5.2-1
Gcc 3.0.4
Atlas 3.6.0-16
4 nodos (8 procesadores).

Dimensión de la matriz	Procesadores	MFLOPS
100	8	414.76
500	8	235.67
1000	8	230.35
2000	8	238.76

Para la medición del rendimiento gráfico se utilizó SPecGLperf que es un programa que mide el desempeño gráfico de un sistema empleando el API de OpenGL. La métrica obtenida es el número de primitivas (punto, línea,

circunferencia, etc) dibujadas por minuto. Así, los resultados obtenidos al ejecutar el programa CopyPixels de SpecGLPerf fueron:

Resolución	**Primitivas por segundo (pps)**
16 x 16	72 200 000
32 x 32	292 600 000
64 x 64	493 000 000
128 x 128	584 000 000
256 x 256	654 600 000
512 x 512	829 000 000

Los resultados obtenidos de las pruebas realizadas nos pueden dar comparación con otros sistemas, sin embargo, es necesario tener en cuenta varios aspectos siendo el más importante el número de procesadores del cluster en cuestión. Si bien podemos comparar nuestro cluster con cualquier otro, se hace necesario hacer dicha comparativa se lleve a cabo con un cluster o máquina de características similares por lo que por el momento no es posible realizar dicha comparativa.