ANÁLISIS E IMPLEMENTACIÓN DE UNA INFRAESTRUCTURA DE CÁLCULO DISTRIBUIDO EN LA RED UNIVERSITARIA

CRISTIAN CAMILO RUIZ SANABRIA

INGENERIA DE SISTEMAS
ESCUELA DE INGENERIA DE SISTEMAS E INFORMÁTICA
FACULTAD DE INGENERIAS FISICO-MECANICAS
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA 2009

ANÁLISIS E IMPLEMENTACIÓN DE UNA INFRAESTRUCTURA DE CÁLCULO DISTRIBUIDO EN LA RED UNIVERSITARIA

AUTOR

CRISTIAN CAMILO RUIZ SANABRIA

Trabajo de grado para optar por el título de ingeniero sistemas

DIRECTOR

Ph.D. JORGE LUIS CHACON VELASCO

CODIRECTOR

ING. ERICK MENESES CUADROS

ASESOR

Ph.D. CARLOS BARRIOS HERNANDEZ

INGENERÍA DE SISTEMAS
ESCUELA DE INGENERIA DE SISTEMAS E INFORMÁTICA
FACULTAD DE INGENERIAS FISICO-MECANICAS
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA 2009

A dios por brindarme salud, sabiduría y permitir que todo se llevara a cabo de la mejor manera.

A mis padres Ana Sanabria y Víctor Ruiz por su amor y compresión, a ellos quienes siempre estuvieron ahí para apoyarme e hicieron todo lo posible para brindarme el estudio

A mi hermano William Ruiz quien siempre creyó en mí y me apoyo en todo momento

A mis amigos de carrera, los cuales aportaron sus valiosos consejos.

Cristian Camilo Ruiz Sanabria

AGRADECIMIENTOS

El autor del proyecto desea expresar sus más sinceros agradecimientos:

Al **Dr Jorge Luis Chacón**, Director del proyecto por su gestión y apoyo durante todo el proyecto.

Al **Ing Erick Meneses**, Codirector del proyecto quien a pesar de sus ocupaciones, siempre estuvo pendiente, aportando su conocimiento y apoyo, sin el cual el proyecto no hubiera podido llevarse a cabo.

Al **Dr Carlos Jaime Barrios**, investigador laboratorio de informática de Grenoble (LIG), por su colaboración con múltiples aspectos del proyecto desde la distancia y su constante apoyo para llevar a cabo un buen proyecto.

Al **Dr Yiannis Georgiou**, investigador laboratorio de informática de Grenoble (LIG), quien siempre estuvo disponible para responder cualquier pregunta y su gran interés en el proyecto.

A la Universidad industrial de Santander y en especial al CENTIC, por proveer los recursos necesarios para la realización del proyecto.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

RESUMEN

Titulo: ANÁLISIS E IMPLEMENTACIÓN DE UNA INFRAESTRUCTURA DE CÁLCULO DISTRIBUIDO EN LA RED UNIVERSITARIA[1]

Autor: CRISTIAN CAMILO RUIZ SANABRIA [**]

Palabras clave: Computación Cluster, Diskless, Explotación de recursos ociosos, Procesamiento distribuido y paralelo.

Descripción: Los clusters computacionales han llegado para aliviar a un costo muy razonable la demanda de cómputo que requieren las investigaciones de hoy día. Aunque su costo es muy pequeño en comparación al costo de las supercomputadoras, el poder de procesamiento que estos proveen es directamente proporcional al número de recursos computacionales que se dediquen para tal fin. Es por esto que se ha desarrollado nuevas alternativas para solventar este problema, la idea principal es poder utilizar recursos que no son usados a plenitud.

Comúnmente las organizaciones cuentan con gran cantidad de recursos de cómputo dedicados a diferentes labores, muchos de estos recursos son solo utilizados durante sus tiempos laborales, y pasan el tiempo restante (horarios nocturnos, vacaciones, fines de semana) completamente inactivos. Algunas implementaciones se han desarrollado con el objetivo de aprovechar estos tiempos muertos de procesamiento, como Computemode, DRBL y I-Cluster.

El presente trabajo pretende desarrollar una solución que pueda aprovechar este tiempo de procesamiento desperdiciado, integrando algunos elementos de la infraestructura de cómputo Computemode en una nueva implementación, basada en el uso de la tecnología ramfs usada por los ramdisk, la cual permite almacenar un sistema de archivos raíz completamente en memoria RAM; con el objetivo de proveer un ambiente diskless que proporcione buen rendimiento en la ejecución de aplicaciones paralelas, así como la capacidad de desplegarse sin ocasionar ningún cambio en las computadoras cliente y con poca configuración por parte del usuario.

---

[1] Trabajo de Grado en la Modalidad de Investigación

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Ph.D Jorge Luis Chacón Velasco. Codirector: Ing. Erick Menéses. Asesor: PhD Carlos Barrios Hernandez

ABSTRACT

TITLE: ANALISYS AND IMPLEMENTATION OF A DISTRIBUTED COMPUTING INFRAESTRUCTURE IN THE UNIVERSITY NETWORK.[*]

AUTHOR: CRISTIAN CAMILO RUIZ SANABRIA[**]

KEYWORD: Clustering Computing, Diskless, resource idleness exploitation, Distributed and parallel processing.

Description: The computational Clusters have reduced to a reasonable cost the demand of computing that the current researches require. Although, its cost is lower than the cost of a supercomputer, the processing power is directly proportional to the number of computational resources dedicated to it. This is the reason why new alternatives have been developed to solve this problem; the main idea is the possibility of using resources that are not being used completely.

Commonly, organizations have a huge number of computational resources dedicated to different activities, many of them are only being used during his labor times and in the remain time they stay inactive. Some implementations have been developing with the objective to take advantage of these idle times like Computemode, DRBL, I-Cluster, etc.

The present work aims to develop a solution that can take advantage of this idle time, integrating some elements of the Computemode computing infrastructure into a new implementation, based on the use of the initramfs technology used for ramdisks, which allow storing a root file system completely in RAM. This propose has the objective of creating a diskless environment that provide a good performance of parallel applications execution, as well as the capacity of doing a deploy without any change in the client`s computer and with few configuration required on behalf of the user.

---

[*] Graduation paper, Research mode.

[**] Faculty of Physical-Mechanical Engineering, School of Engineering and Computer Science. Director: Ph.D Jorge Luis Chacón Velasco. Codirector: Ing. Erick Meneses. Asesor: PhD Carlos Barrios Hernandez

GLOSARIO

***DHCP**:* (sigla en inglés de Dynamic Host Configuration Protocol – Protocolo de Configuración Dinámica de Anfitrión) es un protocolo de red que permite a los nodos de una red IP obtener sus parámetros de configuración automáticamente.

TFTP: Trivial File Transfer Protocol (Protocolo de transferencia de archivos trivial). Protocolo de transferencia muy simple semejante a una versión básica de FTP. TFTP a menudo se utiliza para transferir pequeños archivos entre ordenadores en una red.

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

Kernel: Parte fundamental de un sistema operativo. Es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma más básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema

NIS: Network Information Service (conocido por su acrónimo NIS, que español significa *Sistema de Información de Red*), es el nombre de un protocolo de servicios de directorios cliente-servidor desarrollado por Sun Microsystems para el envío de datos de configuración en sistemas distribuidos tales como nombres de usuarios y hosts entre computadoras sobre una red.

***POST**:* Acrónimo inglés de Power On Self Test (Auto diagnóstico al encender). Es un proceso de verificación e inicialización de los componentes de entrada y salida en un sistema de cómputo que se encarga de configurar y diagnosticar el estado del hardware.

PXELINUX: derivado de SYSLINUX para cargar Linux desde un Servidor en la red local.

RAMDISK: Parte de la memoria principal del computador que ha sido configurada para parecer a la vista de los programas, como si fuese una unidad de disco, aunque muy rápida.

NFS: Network File System *(Sistema de archivos de red)*, o NFS, es un protocolo de nivel de aplicación, según el Modelo OSI. Es utilizado para sistemas de archivos distribuido en un entorno de red de computadoras de área local. Posibilita que distintos sistemas conectados a una misma red accedan a ficheros remotos como si se tratara de locales.

PXE: Hace referencia al entorno de ejecución de pre arranque (Preboot eXecution Environment) [3] es un entorno para arrancar e instalar el sistema operativo en computadores desde una red. PXE está relacionado con varios protocolos de red como IP, UDP, DHCP y TFTP.

ARQUITECTURA: Estructura del sistema; el conjunto de decisiones significativas acerca de cuáles son los componentes software y como deben ir organizados. La selección de los elementos estructurales y sus interfaces de los cuales el sistema está compuesto, junto con su comportamiento[2].

INFRAESTRUCTURA: Estructura fundamental de una organización o sistema. Lo básico, arquitectura fundamental de cualquier sistema (electrónico, mecánico, social, político, etc.) determina como funciona y cuan flexible es para satisfacer futuros requerimientos.[3]

---

[2] Robert V, Lavatte C. Objet-Oriented systems analysis and design with UML. Prentice Hall 2005.

[3] Infrastructure Definition. Enciclopedia PC Magazine (Julio, 2009). Disponible en: http://www.pcmag.com/encyclopedia_term/0,2542,t=infrastructure&i=44985,00.asp

1. INTRODUCCIÓN

Los proyectos de investigación en la actualidad están requiriendo cada vez más, mayores capacidades de cómputo para tratar los problemas que afrontan. Estos requerimientos computacionales son típicos en aplicaciones como algoritmos genéticos, bases de datos, tratamiento de imágenes, simulación, dinámica de fluidos, química cuántica, biomedicina, etc. En un principio los sistemas multiprocesador solo eran implementados en supercomputadoras, y debido a su alto costo tenían un uso limitado siendo inaccesibles para proyectos académicos. Gracias al desarrollo de los sistemas operativos, la supercomputación ofrece una nueva perspectiva, ya que con los denominados Clusters se permite la integración de maquinas independientes, interconectándolas en un solo conjunto, dando al programador y al usuario la apariencia de un solo supercomputador.

Nuestro entorno local se caracteriza por poseer recursos bastante limitados para la ejecución de las investigaciones, lo que haría difícil la adquisición de computadoras más sofisticadas como pueden ser "mainframes" o supercomputadoras. Por lo tanto los clusters constituirían una de las alternativas que más se ajusta a nuestro entorno ya que brindaría a las investigaciones capacidades de cómputo considerables, aprovechando de una mejor manera los recursos existentes sin la necesidad de grandes inversiones.

Recientemente en la Universidad Industrial de Santander se han venido realizando varios proyectos, en áreas como el tratamiento de imágenes y la inteligencia artificial por mencionar algunas, que se caracterizan por la necesidad de realizar gran cantidad de cálculos para el procesamiento de los datos. Muchas veces este procesamiento debe realizarse en un tiempo corto o en los casos más críticos en tiempo real, lo que provoca que se adopte enfoques de programación paralela en su implementación. El personal a cargo del desarrollo de estos proyectos tiene que preocuparse por conseguir los recursos computacionales, al igual que del montaje de la infraestructura computacional en la que se va a ejecutar la aplicación, debido a que

actualmente en la universidad no existe ninguna infraestructura de cómputo distribuido. Malgastándose tiempo y desviando la atención del objetivo principal, que es el desarrollo de la aplicación y no la implementación de la infraestructura.

El presente proyecto busca brindar un prototipo de una infraestructura de cómputo distribuido integrando recursos existentes en su tiempo ocioso de una manera fácil y transparente para el usuario, con el objetivo de que pueda ser utilizada para el desarrollo, prueba, y ejecución de aplicaciones en el campo de la computación distribuida o paralela. El resto del documento está organizado de la siguiente manera: en la sección 2 se realiza el planteamiento del problema y se muestran los objetivos de este trabajo, en la sección 3 se expone el marco teórico donde se establece el contexto y se presentan los diferentes paradigmas de la computación, seguidamente se hace una revisión del estado del arte. Luego en la sección 5 se desarrolla la propuesta donde se muestra el diseño seguido de la implementación de un prototipo con su respectiva evaluación que permite evidenciar las posibilidades del modelo propuesto. Finalmente se presentan las conclusiones en la sección 7.

2. PLANTEAMIENTO DEL PROBLEMA Y PROPUESTA

2.1 PLANTEAMIENTO DEL PROBLEMA

En los últimos años la disponibilidad de microprocesadores económicos de alto rendimiento y redes de alta velocidad, han provocado que el personal académico de diversas unidades y centros de investigación, se hayan dado a la tarea de construir sus propias supercomputadoras conectando computadoras personales y desarrollando software para enfrentar tales problemas. Aunque esta solución a reducido enormemente los costos de tener a disposición gran potencia computacional, la potencia que ofrecen es proporcional a la cantidad de recursos de computo dedicados para tal fin, y dado que esta depende de los fondos del grupo investigador, nunca se llega a conseguir el objetivo perseguido, lo cual causa que algunos centros de investigación vean limitada su capacidad de computo por la disponibilidad de recursos. Otro problema que se ve con frecuencia en estos tipos de sistemas, ocurre cuando el número de nodos crece, la administración de su software como la actualización de este introduce labores tediosas para los administradores.

Comúnmente en los centros de investigación, en las universidades o empresas, existen gran cantidad de recursos de computo que no se están aprovechando en su totalidad, debido a diferentes políticas que impiden que se modifique o instale alguna clase de software (necesario para poder realizar algún tipo de computo distribuido) sobre las máquinas de la organización. Esto da como resultado que se desperdicie el poder de computo que estas máquinas podrían brindar en los momentos en que están inactivas, como puede ser en horarios nocturnos, días festivos, vacaciones del personal etc.

Este panorama se vive en la Universidad Industrial de Santander donde se cuenta con una considerable cantidad de equipos de cómputo, los cuales son utilizados para diferentes actividades académicas, al igual que para uso individual de los estudiantes. Estos recursos solo pueden usarse en horarios restringidos, a lo cual se suma la imposibilidad de

instalar algún tipo de software en ellos, dando como resultado que no sea factible integrarlos a alguna infraestructura de cómputo con implementaciones tradicionales. Consientes de esta problemática y del numero de computadores personales que hay distribuidos en las aulas de informática de la universidad, se hace necesario pensar en una estrategia diferente, la cual pueda aprovechar los recursos existentes, en lugar de inversiones en otros nuevos, se plantea la posibilidad de unificar todos los recursos, de manera que también puedan ser útiles a la comunidad científica.

2.2 DESCRIPCION DEL AMBIENTE

La universidad industrial de Santander cuenta con una gran cantidad de recursos de cómputo distribuidos en salas de informática ubicadas en distintos edificios del campus universitario. Entre estas salas se pueden nombrar salas de informática pertenecientes a la escuela de sistemas, las salas de cómputo que están disponibles para actividades académicas y uso de los estudiantes en el CENTIC, que es quizás el edificio con más concentración de equipos de cómputo presente en la universidad. Estos recursos destinados principalmente al estudiantado presentan tiempos ociosos mientras no están al servicio del estudiante como son en el horario de 8 pm a 6 am, además del día domingo en el cual permanecen completamente inactivos.

Cada sitio maneja los recursos con sus políticas y generalmente se hace difícil la instalación de cualquier tipo de software, si logra ser instalado no se tiene la certeza de que este software permanecerá, esto impediría cualquier intento de conformar algún tipo de infraestructura de cálculo con estos equipos que utilizara sus tiempos de ocio. Otro factor que se suma a los anteriores es la presencia del sistema operativo Windows en la mayoría de los recursos antes citados lo cual hace necesario el uso de otro sistema ya que las implementaciones de cálculo que se ejecutan paralelamente o distribuida mente generalmente están implementadas y se desarrollan en plataformas Linux.

2.3 REQUERIMIENTOS

- Causar el mínimo impacto en las actividades diarias de estas salas de cómputo.

- No depender de software instalado en las máquinas.
- Bridar al usuario de la infraestructura de cálculo, un buen rendimiento en la ejecución distribuida y paralela de software, además de un acceso transparente a la infraestructura.

2.4 PROPUESTA DE SOLUCIÓN

1.1.1 OBJETIVOS

1.1.1.1 Objetivo general

Implementar una arquitectura distribuida que haga uso de los recursos de cómputo existentes en la red universitaria durante su tiempo ocioso, con el fin de obtener una infraestructura que proporcione alto poder de cálculo.

1.1.1.2 Objetivos específicos

- Diseñar una arquitectura distribuida, que permita el monitoreo, administración y despliegue de sistemas operativos Linux a través de la red universitaria; Teniendo en cuenta factores como: rendimiento, ubicación y disponibilidad de recursos de cómputo.
- Seleccionar y desarrollar las herramientas tecnológicas necesarias (protocolos, servicios, módulos, etc.) para la implementación de un prototipo de la arquitectura propuesta, que una vez instaladas, configuradas, e integradas satisfaga los requerimientos del diseño planteado.
- Implementar una interfaz de administración mediante la instalación de software existente y desarrollado, integrando herramientas de monitoreo y administración de recursos; de tal manera que se ajuste a las necesidades del diseño y permita a los usuarios una interacción fácil y eficaz con el sistema.
- Evaluar el prototipo implementado a través de la realización de diferentes pruebas, que permitan medir la funcionalidad del sistema, así como el rendimiento que tiene este, en el proceso

de integración de recursos y en la ejecución de trabajos generados por diferentes usuarios del sistema.

3. MARCO TEORICO

3.1 COMPUTACIÓN CONCURRENTE, DISTRIBUIDA Y PARALELA

La computación Concurrente es una forma de computación en la cual los programas son diseñados como un conjunto de procesos que se comunican entre sí. Los programas concurrentes pueden ser ejecutados secuencialmente en un solo procesador intercambiando la ejecución de un determinado proceso en distintos instantes del programa o pueden ser ejecutados en varias máquinas asignando cada proceso a una máquina especifica, estas máquinas pueden estar cerca o estar distribuidas a través de la red. Es aquí cuando se puede hablar de computación distribuida y computación paralela donde la principal diferencia que hay entre las dos formas de computación es la simultaneidad que hay entre su ejecución, pues en la computación paralela todo los procesos se ejecutan al mismo tiempo, lo cual no es necesariamente el caso de la computación distribuida, además de esto, comúnmente la computación paralela utiliza recursos que están ubicados en un mismo sitio en cambio la computación distribuida puede utilizar recursos ubicados en diferentes sitios separados geográficamente.

"*A pesar de que la Ley de Moore sigue vigente y los procesadores doblan su potencia cada 18 meses, los ordenadores no son capaces de absorber la demanda de almacenamiento y procesamiento de datos que genera la investigación en la actualidad*"[4]. La computación paralela y distribuida se ha venido utilizando para aliviar la insuficiencia de equipo tecnológico. Estas pueden ser utilizadas para permitir agregar y coordinar diferentes recursos de cómputo con el fin de generar una gran capacidad de cálculo.

[4] Nacho Rojo. El PC al servicio de grandes causas (2004, octubre). Disponible en: http://www.consumer.es/web/es/tecnologia/internet/2004/10/11/110178.php

3.2 SISTEMA DISTRIBUIDO

Los recursos necesarios para el procesamiento de los datos producto de las investigaciones, no siempre están ubicados en un mismo sitio por razones económicas de espacio físico, seguridad o infraestructura. Además en muchas ocasiones los recursos computacionales con que cuenta una organización, suelen estar subutilizados. Para lograr una mejor utilización de los recursos propios y ajenos en busca de satisfacer las grandes necesidades de procesamiento y almacenamiento de datos de las organizaciones y en especial de los investigadores, se ha venido dando origen a un nuevo concepto llamado computación distribuida la cual constituye una solución que consiste en distribuir el trabajo entre multitud de ordenadores conectados, aprovechando la gran cantidad de tiempo en que éstos no se utilizan.

Un sistema distribuido se define entonces como una colección de computadores separados físicamente y conectados entre sí por una red de telecomunicaciones distribuida; cada máquina posee sus componentes de hardware y software que el usuario percibe como un solo sistema (no necesita saber qué cosas están en que máquinas). El usuario accede a los recursos remotos de la misma manera que accede a recursos locales.

Características

- Para cada uno de los usuarios debe de ser similar al sistema centralizado.
- Se ejecuta en múltiples computadoras.
- Tiene varias copias del mismo sistema operativo o varios sistemas operativos que ofrecen los mismos servicios.
- Transparencia (El uso de múltiples procesadores y el acceso remoto debe de ser invisible)

La aplicación de estos conceptos busca resolver problemas existentes en la computación como:

Potencia de cálculo: La unión de varios computadores no será superada por una supercomputadora, es decir que aunque la supercomputadora sea muy robusta no igualara la unión de los recursos de muchos computadores que al unirse cada vez más incrementan su potencial y semejaran una supercomputadora.

Tiempo: La computación distribuida permite que investigaciones o procesos que requieren manejo y procesamiento de gran cantidad de información se haga de forma más rápida, lo que permite obtener resultados en menos tiempo.

Almacenamiento: Al reducir tiempo en el procesamiento de la información se incrementa la cantidad de datos que se obtienen, estos datos no necesariamente tiene que ser almacenados en una sola máquina, debido a que su tamaño obligaría a tener una gran infraestructura de almacenamiento aumentando los gastos y tiempos de acceso.

Dinero: Debido a los altos costos, no todas las organizaciones tienen la capacidad de adquirir maquinas robustas con configuraciones de alto desempeño que les permitan asumir proyectos que requieran este tipo de computadoras. Mediante la computación distribuida las empresas tienen acceso a muchos computadores que son subutilizados y que al interconectarse pueden llegar a tener características superiores a las de un supercomputador, ahorrando costos en la adquisición de equipos, mantenimiento, infraestructura y administración.

3.3 ARQUITECTURAS DISTRIBUIDAS

3.3.1 CLUSTERS

Un cluster es un tipo de sistema de procesamiento paralelo o distribuido, el cual consiste de una colección de computadoras individuales interconectadas trabajando juntas como un solo recurso de cómputo. Cada sistema estará suministrando un mismo servicio o ejecutando una (o parte de una) misma aplicación paralela. La característica inherente de un cluster es la de compartir recursos: ciclos

de CPU, memoria, datos y servicios. Los clusters están conformados por computadores genéricos con uno o más procesadores, denominados nodos.

3.3.1.1 Clasificación de los clusters

El termino cluster tiene diferentes connotaciones para diferentes grupos de personas. Los tipos de clusters, según el uso que se les dé y los servicios que ofrezcan, determinan el significado del término para el grupo que lo utiliza. Tres distinciones básicas en los cluster son la base para su clasificación, el rendimiento, el balanceo de carga y la disponibilidad; si bien todas se pueden lograr con un cluster al optimizar una se pierden características de las otras, por lo tanto, según la aplicabilidad de los clusters se han desarrollado diferentes líneas tecnológicas.

Alto rendimiento: Esta línea surge frente a la necesidad de súper computación para determinadas aplicaciones, lo que se persigue es conseguir que un gran número de maquinas individuales actúen como una sola maquina muy potente, este tipo de cluster se aplica mejor en problemas grandes y complejos que requieren una cantidad enorme de potencia computacional. Este tipo de clusters en general está enfocado hacia las tareas que requieren gran poder computacional, grandes cantidades de memoria, o ambos a la vez, teniendo en cuenta que las tareas podrían comprometer los recursos por largos periodos de tiempo.

Balanceo de carga: Este tipo de tecnología de clusters, es el destinado al balanceo de carga. Surge el concepto de clusters de servidores virtuales, lo cual es un cluster que permite que un conjunto de servidores de red compartan la carga de trabajo y de tráfico de sus clientes, aunque aparezcan para estos clientes como un único servidor. Al balancear la carga de trabajo en un conjunto de servidores, se mejora el tiempo de acceso y la confiabilidad. Además, como es un conjunto de servidores el que atiende el trabajo, la caída de uno de ellos no ocasiona la caída total del sistema. Este tipo de servicio es de gran valor para las compañías que trabajan con grandes volúmenes de tráfico y trabajo en sus web.

Alta disponibilidad: El objetivo aquí es la máxima disponibilidad de servicios y el rendimiento sostenido lo cual se puede lograr con el mantenimiento de servidores que actúen entre ellos como respaldos de la información que sirven. La flexibilidad y robustez que proporcionan este tipo de clusters, los hace necesarios en ambientes de intercambio masivo de información, almacenamiento de datos sensibles y allí donde sea necesaria una disponibilidad continua del servicio ofrecido.

3.3.2 GRID

La grid permite el intercambio, selección y agregación de una amplia variedad de recursos entre los que se pueden nombrar: computadoras, supercomputadoras, sistemas de almacenamiento, dispositivos especializados, etc. Que están generalmente distribuidos geográficamente y pertenecen a diferentes organizaciones, con el fin de resolver problemas que impliquen cómputo a gran escala y uso intensivo de datos en la ciencia, industria y el comercio. En [10] se nombran los diferentes servicios que puede proveer una grid computacional.

Una de las principales diferencias entre la computación grid y la computación cluster es la forma en la que los recursos son administrados, en el caso de los clusters la asignación de los recursos es llevada a cabo centralizadamente por un administrador de recursos, y todos estos recursos trabajan juntos cooperativamente como un solo recurso unificado en cambio, en el caso de la grid la administración tiende a ser mas descentralizada y no intenta proveer una sola imagen del sistema[5].

3.3.3 COMPUTACIÓN VOLUNTARIA O DE CICLOS REDUNDANTES

Es un tipo de computación distribuida en el cual los dueños de las computadoras donan sus recursos computacionales (como el poder de procesamiento o almacenamiento) para unos o mas proyectos. Consiste en un servidor o grupo de servidores que distribuyen trabajo de procesamiento a un grupo de computadoras voluntarias a ceder

[5] Grid Computing Info Centre, http://www.gridcomputing.com/gridfaq.html

capacidad de procesamiento no utilizada. Básicamente, cuando se deja el computador encendido, pero sin utilizarlo, la capacidad de procesamiento se desperdicia por lo general en algún protector de pantalla, este tipo de procesamiento distribuido utiliza nuestra computadora cuando nosotros no la necesitamos, aprovechando al máximo la capacidad de procesamiento.

Algunos proyectos mundiales en los cuales se emplea la computación voluntaria son: Foldin@Home[6], LHC@Home[7], SETI@Home[8] etc.

3.4 DISKLESS CLUSTER

Con el fin de mejorar la administración de los Clusters y también reducir los costos se ha recurrido a los llamados Diskless Cluster, en estos tipos de Clusters la principal característica es que no poseen disco o su disco local no es utilizado en el cómputo. Se basan en un almacenamiento centralizado reduciendo tareas administrativas ya que todos los nodos utilizan la misma imagen de sistema operativo que reside en un servidor. Algunas de las ventajas en utilizar Clusters sin disco, además de las ventajas administrativas son:

1. Reducción de costos y de fallas en los nodos.
2. Posibilidad de incorporación de recursos no dedicados.

Los clusters construidos para aplicaciones científicas, particularmente aplicaciones paralelas, tienen poca necesidad de discos locales. Una vez los nodos son iniciados, una aplicación puede ser distribuida a los nodos destino en una variedad de formas, las cuales ninguna tiene necesidad de un disco local.

Solo se utilizan recursos como CPU y memoria. El acceso de archivos por medio de la red, puede en algunos casos disminuir la latencia, en comparación al acceso a un disco. Lo que sí está prohibido en estos tipos de Cluster es la memoria virtual, ya que localmente no se puede

[6] Foldin@home, Understaning de protein folding, http://folding.stanford.edu/
[7] LHC@Home, LHC at home, http://lhcathome.cern.ch/
[8] Seti@Home, Detección de señales de otras civilizaciones, http://seti.astroseti.org/setiathome/

acceder a ningún tipo de unidad de disco, la memoria virtual se implementaría desde red en un espacio físico en disco que reside en el servidor, esto degradaría seriamente el desempeño del Cluster.

3.4.1 CARGA DE UN SISTEMA LINUX

3.4.1.1 INICIALIZACION DE UN SISTEMA LINUX A NIVEL DE SERVICIOS

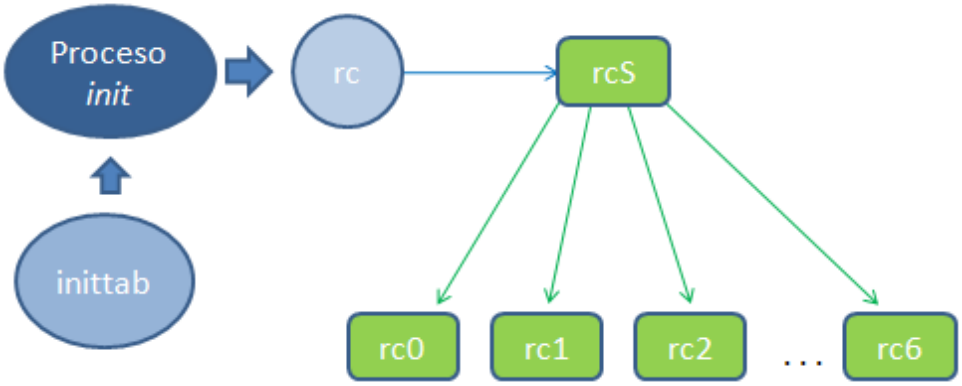

Ilustración 1 Inicialización de un sistema Linux a nivel de servicios

En la figura se observa el proceso de carga de los servicios en Linux, el primero proceso en espacio de usuario en ejecutarse es el proceso *init*, este es el padre de todos los procesos, dicho proceso es el encargado de inicializar los demás servicios que permitirán la carga completa de todo el sistema Linux. El proceso *init* lee el archivo *inittab* en el cual esta especificado el nivel de ejecución a iniciar, un nivel de ejecución se define como un conjunto de servicios a ser inicializados. Luego de determinar el nivel de ejecución, el proceso *init* ejecuta el script *rc* el cual se encarga de inicializar los servicios del nivel de ejecución respectivo, todos los niveles ejecutan un nivel básico, el cual contiene todos los servicios elementales para la inicialización de los servicios de cada nivel.

3.4.1.2 SISTEMA LINUX EN DISCO

1. La carga de un sistema operativo Linux empieza con el inicio de la computadora, esta ejecuta el POST el cual realiza un proceso de revisión de todo el hardware presente en la computadora. Luego la BIOS se encarga de traer a memoria los primeros 512 Kbytes que constituye el primer sector del disco duro, el cual contiene la primera fase del cargador de arranque, seguidamente presenta al usuario un menú en el cual muestra los sistemas disponibles para arrancar, una vez el usuario ha hecho su elección, este carga el kernel Linux en memoria, el cual empieza a ejecutarse.

2. Con el fin de reconocer la mayor cantidad de hardware posible sin hacer el kernel Linux tan pesado, este viene acompañado de un pequeño sistema de archivos comprimido llamado ramdisk que es montado en memoria por el kernel y el cual le permite cargar los módulos necesarios para el funcionamiento, además de realizar algunas configuraciones que permitan el montaje del verdadero sistema de archivos raíz que normalmente estará en disco.

3. Luego de que se han ejecutado todos los scripts presentes en la ramdisk y se a montado el verdadero sistema de archivos, se ejecuta el primer proceso de usuario que se conoce como init, este se encarga de iniciar los servicios restantes para así culminar con la carga del sistema operativo.

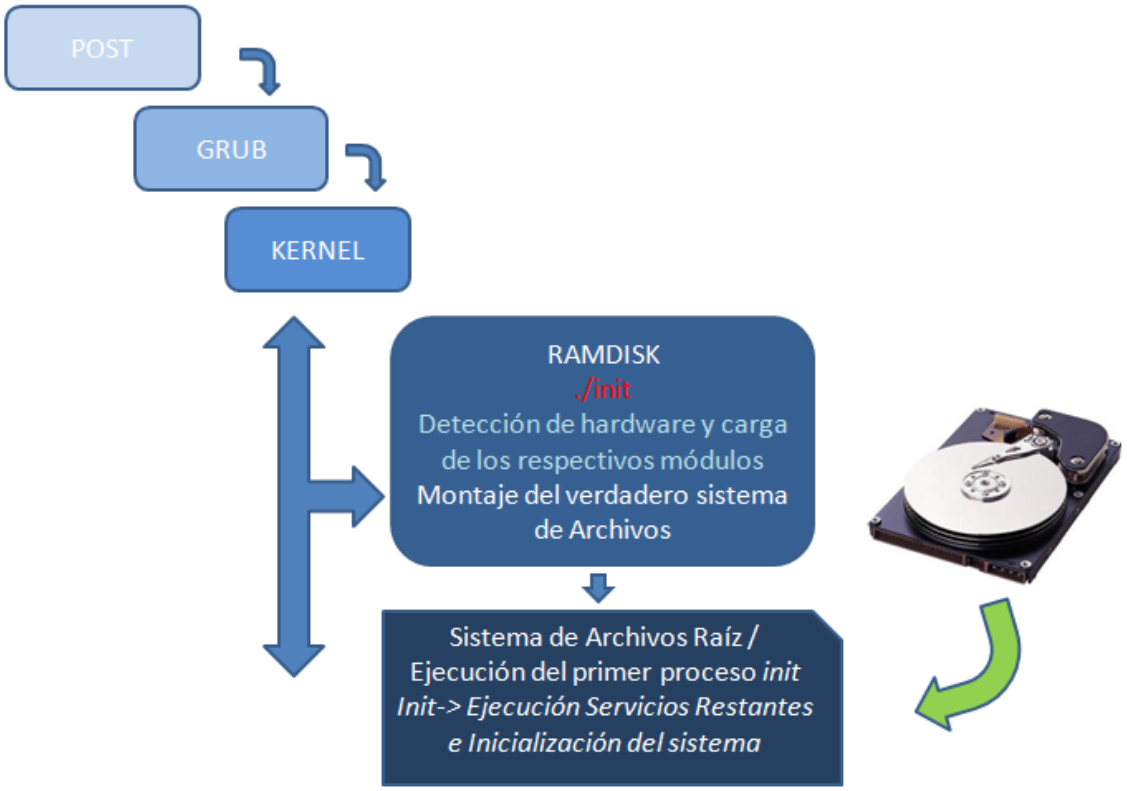

Ilustración 2 Procedimiento de Carga de un sistema Linux en disco

3.4.1.3 UN SISTEMA LINUX EN MEMORIA

A continuación se describe el proceso de carga de un sistema en una infraestructura diskless.

1. Se realiza una petición por parte de la tarjeta de red del equipo, de una dirección y otros parámetros de configuración a un servidor que está a la escucha de estas peticiones.
2. Una vez obtenidos los parámetros, se descarga de un servidor TFTP un gestor de arranque para sistemas operativos.
3. Descargado el gestor, este toma el control y hace otra petición a un servidor TFTP, que se encarga de proveerle el Kernel.
4. Una vez obtenido este, se inicia y monta su sistema de archivos principal valiéndose de un sistema de archivos de red.

Lo anterior se pude ilustrar de la siguiente manera:

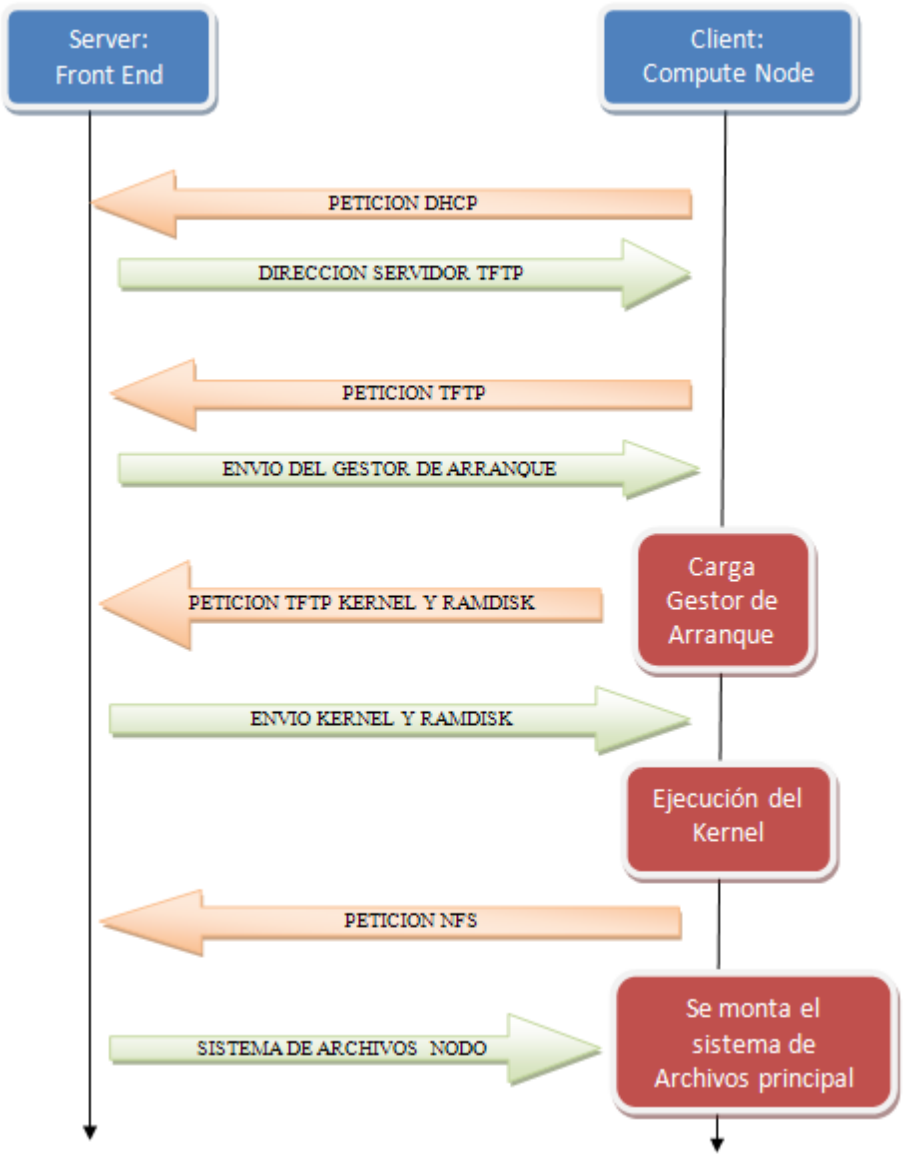

Ilustración 3 Procedimiento de carga de un sistema diskless

3.4.1.4 MODOS DE IMPLEMENTAR EL SISTEMA DE ARCHIVOS EN UN SISTEMA DISKLESS

Para el sistema de archivos raíz se consideran tres técnicas para su implementación:

- Crear un sistema de archivos que resida totalmente en RAM, esto se conoce como RAMDISK. Son muy utilizados en los sistemas operativos modernos para configuraciones previas, antes de montar el sistema de archivos principal. Se reserva cierta cantidad de espacio en memoria para poder albergar el sistema de archivos completo. Esta técnica tiene algunas desventajas como la probabilidad que tiene el sistema operativo de quedarse sin RAM para poder ejecutar sus aplicaciones, ya que estas porciones que se reservan son estáticas, y se tiene que elegir un espacio en RAM que pueda contener todo el sistema de ficheros. Entre las ventajas se puede nombrar la disponibilidad de todos los archivos, comandos y datos del usuario, residentes en memoria proporcionando mas desempeño.
- Implementar el sistema de archivos totalmente en red, esto se lleva a cabo compilando el Kernel con soporte para montar el sistema de archivos por red, además se deben compilar otras características como el driver de la tarjeta de red, y configuración IP por medio de DHCP. Esto trae la ventaja de disponer de más memoria RAM para la ejecución de los programas por parte del usuario. La desventaja radica en generar más tráfico en la red, y aumentar un poco la latencia en la ejecución de las aplicaciones.
- Combinar las dos técnicas anteriores, se utiliza un RAMDISK para las configuraciones iníciales, tales como cargar los módulos para la tarjeta de red, creación de algunos archivos de configuración, y luego se procede a montar el sistema de archivos principal por medio de la red. Esto trae algunas ventajas como la fácil creación y configuración de sistemas operativos

para cada nodo y la no necesidad de compilar un Kernel con características especiales.

3.4.2 SISTEMAS DE ARCHIVOS QUE SE ALMACENAN EN RAM

3.4.2.1 TMPFS

Sistema que almacena todo en la cache de disco creada por el kernel [11], crece y divide el almacenamiento con el fin de acomodarlo en la memoria existente, adicionalmente cuenta con la capacidad de enviar las páginas no usadas a la memoria swap. Tiene un tamaño límite máximo pero este puede modificarse por medio del siguiente comando 'mount -o remount ...'. En comparación con el sistema de archivos ramfs, se gana posibilidad de acceder a la swap además de chequeo del límite de memoria.

3.4.2.2 RAMFS

Es un sistema de archivos muy simple que utiliza la cache de disco creada por el kernel Linux como un sistema de archivos dinámico el cual puede cambiar su tamaño. Normalmente todos los archivos son almacenados temporalmente en la memoria por Linux, paginas de datos son mantenidas en caso de ser necesitadas de nuevo, pero son marcadas como libres, en caso de que el sistema de memoria virtual necesite memoria para alguna otra cosa. Similarmente, los datos escritos en archivos son marcados como libres tan pronto son escritos en algún tipo de almacenamiento, pero mantenidos en memoria en caso de ser necesitados de nuevo hasta que el sistema de memoria virtual decida reasignar la memoria.

Con el sistema de archivos ramfs, no hay un almacenamiento de respaldo. Archivos escritos en el sistema ramfs se almacenan en páginas como es usual, pero no hay almacenamiento secundario a donde enviarlas. Esto significa que las páginas nunca son marcadas como libres, por lo tanto no pueden ser liberadas por el sistema de memoria virtual cuando está en la búsqueda de memoria libre. La cantidad de código requerido para implementar ramfs es pequeño, porque todo el trabajo ya está hecho por la infraestructura de caching existente en linux. Básicamente, se esta montando la cache de disco como un sistema de archivos.

3.4.2.3 USO DE LOS SISTEMAS DE ARCHIVOS ALMACENADOS EN RAM

Los sistemas de archivos implementados en ram, son utilizados por los sistemas linux principalmente para dos propósitos:

ramfs: en los kernels recientes de la serie 2.6 se utiliza ramfs como sistema de almacenamiento para el disco ram de inicio llamado initramfs.

tmpfs: se utiliza comúnmente para montar algunos directorios importantes en el sistema raíz de linux como son: /var/run, /var/lock, /dev, etc.

4. ESTADO DEL ARTE

Por un lado la dificultad para encontrar recursos computacionales dedicados a procesamiento, y por otro la abundancia de los mismos dedicados a otros oficios donde solo se usa una fracción de su capacidad, han hecho que aumente el interés y estudio en infraestructuras cluster flexibles que puedan aumentar su poder computacional haciendo uso del tiempo de procesamiento ocioso con maquinas de la organización. Dicha propuesta ha sido abordada desde diferentes puntos de vista y el objetivo es claro llegar a usar la misma máquina para dos actividades distintas: las actividades rutinarias de la organización y como un nodo de procesamiento para un cluster. Con el fin de conservar la integridad de la configuración y ciertos componentes del cluster, no es factible que el usuario en su trabajo diario pueda utilizar el mismo espacio de trabajo y sistema, que utiliza el cluster. Además se ha demostrado en [1] que el sistema más utilizado es Windows con más de un 90 % de uso y el ambiente común en las implementaciones cluster es Linux. Esto conlleva a que se necesite dos sistemas completamente separados, uno para el cluster y otro para el usuario común, para lo cual han surgido las siguientes soluciones:

- Alojamiento de los sistemas en dos particiones diferentes que se ejecutan a diferentes turnos.
- Maquinas virtuales.
- Metodología diskless

Para llevar a cabo lo antes dicho hay que tener en cuenta el impacto que esto va a causar al usuario que va a prestar su computadora, se puede utilizar su computadora en horarios en que no se utilice como en horarios nocturnos, o podría aprovecharse oportunidades de procesamiento en su tiempo ocioso.

4.1 ALOJAMIENTO DE SISTEMAS EN PARTICIONES DIFERENTES

Sistemas como I Cluster[2] implementan una solución basada en el reinicio de la maquina motivados por los asuntos de seguridad. Estos sistemas al detectar que no hay actividad del usuario, reinician la maquina entrando en un sistema operativo separado y diferente en el cual la aplicación cliente puede ejecutarse. Esta implementación requiere de una partición independiente para el entorno paralelo, proporcionando un sistema operativo separado al igual que un sistema de ficheros, preservando los datos del usuario. Como ventaja el entorno paralelo puede correr en un sistema operativo diferente al que el usuario habitualmente utiliza, por ejemplo para los sistemas I-Cluster y vCluster los usuarios podrían utilizar Windows mientras que el entorno de programación paralela utilizaría Linux.

Una desventaja de este método es que requiere un reinicio para el cambio entre los dos sistemas operativos y esta operación tiene un impacto sobre el usuario interactivo. Esto se debe a que el cambio entre dos sistemas operativos no es instantáneo, toma decenas de segundos en el mejor de los casos. Para minimizar tal impacto I-Cluster y vCluster hacen un seguimiento de la actividad del usuario para intentar predecir cuando el usuario va a necesitar otra vez su computadora. Esta técnica es usada para evitar reiniciar la máquina en modo Cluster cuando el usuario esperaba utilizarla pronto, así como reiniciarla de nuevo al ambiente del usuario de una manera anticipada.

A nivel local se cuenta con un trabajo realizado en la universidad de Antioquia [12] el cual consistió en la utilización de las salas de computo de la universidad, para implementar un cluster utilizando la herramienta rocks, las máquinas fueron configuradas para tener dos particiones una para el sistema utilitario que en el caso del trabajo fue Windows y otra para el cluster.

4.2 AMBIENTES BASADOS EN MÁQUINAS VIRTUALES

Infraestructura Grid basada en virtualización [13], afronta la disponibilidad de recursos dedicados basado en el uso de máquinas virtuales, la propuesta consiste en utilizar la técnica de paravirtualizacion implementada por Xen, con el fin de tener dos sistemas operativos ejecutándose concurrentemente, uno dedicado a atender al estudiante (usuario local) y otro es utilizado como un nodo de procesamiento que se integra a una infraestructura de grid existente. El mecanismo utilizado brinda ventajas como el checkpointing, la migración en caliente entre otras, manteniendo un nivel de calidad de servicio entre los dos usuarios.

4.3 SOLUCIONES DISKLESS

4.3.1 VENTAJAS AL UTILIZAR DISKLESS

Los sistemas diskless son una interesante opción para la implementación de clusters computacionales por los siguientes motivos: disminuyen el costo total del sistema, reducen las posibilidades de fallas en disco, son más fáciles de administrar, además del hecho de que muchas aplicaciones de cálculo intenso no requieren ninguna unidad local de almacenamiento. Pero quizás la ventaja más significativa de estos sistemas es la posibilidad de integrarlos a infraestructuras de cómputo existentes para la utilización de los recursos ociosos.

4.3.2 ESTUDIO DE LAS SOLUCIONES DISKLESS

Esta revisión de literatura trata de dar respuesta a la escogencia de la herramienta de computo a utilizar, a continuación se describirán cada una de las herramientas encontradas, se expondrán sus ventajas y desventajas.

4.3.2.1 PELICAN

Descripción:

Es una distribución Live Cd basada en debian, cuenta con un conjunto de servicios que le permiten descubrir, agregar e integrar computadoras en la misma red para formar un cluster en pocos minutos. El nodo maestro (fronted) sea una computadora real o una maquina virtual inicia de la imagen del CD. Los nodos de cómputo inician por medio de PXE, usando el nodo maestro como servidor, todos los nodos obtienen sus sistemas de archivos del mismo CD, por lo tanto esta garantizado que todos ejecuten el mismo software, ninguna computadora cliente sufre cambio alguno ya que su sistema de archivos se encuentra alojado en el servidor. Integra varias herramientas para computo paralelo como MPI, PVM, OCTAVE, GROMACS además de software para monitorización y administración de los recursos de un cluster como: Ganglia y SLURM.

Ventajas.

- Es de fácil manejo y configuración, es apropiado para la realización de pruebas ya que es un live cd.
- Provee un entorno de fácil montaje para la compilación de software paralelo.
- Permite crear una versión que se adapte a las necesidades de cada individuo a través de la herramienta Debian live.

Desventajas:

- Es difícil agregar software nuevo, tiene problemas con la integración de paquetes .deb.
- No cuenta con una interface de administración que permita el manejo de recursos no dedicados.
- Presenta conflictos si hay un servidor DHCP funcionando en la red.

4.3.2.2 ONESIS

Descripción:

Permite integrar nodos diskless, nodos diskfull o cualquier combinación de ellos. Montajes simples raramente requieren alguna configuración, con ONESIS incluso ambientes complejos se convierten en ambientes de fácil manejo.

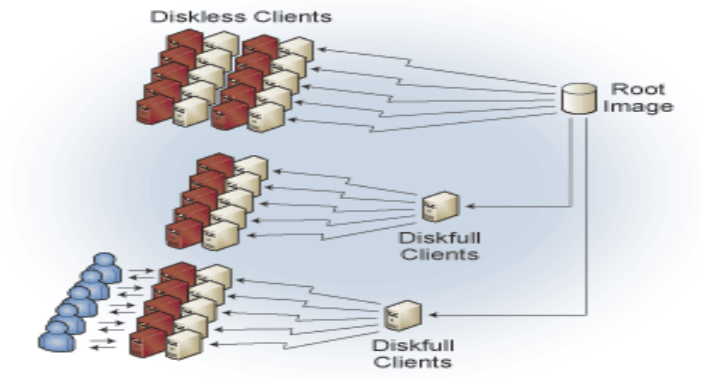

Ilustración 4 Arquitectura de ONESIS

Fuente: http://onesis.sourceforge.net/

¿Que hace ONESIS?

Una sola imagen de ONESIS puede contener la configuración de varias máquinas diferentes al mismo tiempo. De hecho mucho de lo que ONESIS hace es explícitamente definir todas las diferencias que existen entre los nodos. Para ser precisos, ONESIS es capaz de llevar a cabo las siguientes funciones:

- Configuración de una distribución Linux usando un sistema de archivo raíz de solo lectura.
- Configuración de cualquier archivo o directorio que sea modificable en la imagen.
  - En RAM (para datos no persistentes)
  - En un disco local
  - En sistemas de archivos compartidos.

- Declaración de clases específicas o versiones de nodos específicos de cualquier archivo o directorio.
- Despliegue de imágenes a un disco local
- Sincronización de la imagen desplegada en el disco local.

- ONESIS cuenta con varias utilidades que pueden ayudar a simplificar la administración del sistema

Ventajas

- Generación de una Ramdisk inicial (initrd) para iniciar nodos que monten su sistema raíz por NFS.
- Manejo del consumo energético. Reuniendo diferentes utilidades de energía dentro de un solo comando.
- Manejo de la administración por consola. Reuniendo diferentes utilidades de consola dentro de un solo comando.
- Configuración PXE haciendo que esta sea más fácil de manejar nodo por nodo.
- Permite la construcción de imágenes a partir de sistemas preinstalados.

Desventajas

- No está diseñada para el computo hpc, por lo tanto no cuenta desde un principio con ninguna herramienta de hpc.

4.3.2.3 DRBL

Descripción:

Es un sistema libre y de código abierto para el manejo y despliegue de sistemas operativos GNU/Linux a través de muchos clientes. Trabaja con Debian, Ubuntu, Mandriva, RedHat, Fedora, CentOS y SUSE. DRBL usa PXE/etherboot, NFS y NIS para proveer servicios a las máquinas clientes por lo cual no se hace necesaria la instalación individual en los discos de las máquinas, utiliza sistemas de archivos que comparte por red.

Ventajas

- Cuenta con una amplia compatibilidad con sistemas Linux.

- Permite la creación de diferentes sistemas para el despliegue con determinadas configuraciones.
- Permite Cargar sistemas a partir de una sola imagen o a través de diferentes imágenes que se pueden configurar.

Desventajas

- No cuenta con un manejador de colas por defecto.
- No está destinado al computo HPC por tanto por defecto no cuenta con las herramientas necesarias para este fin.

4.3.2.4 COMPUTEMODE

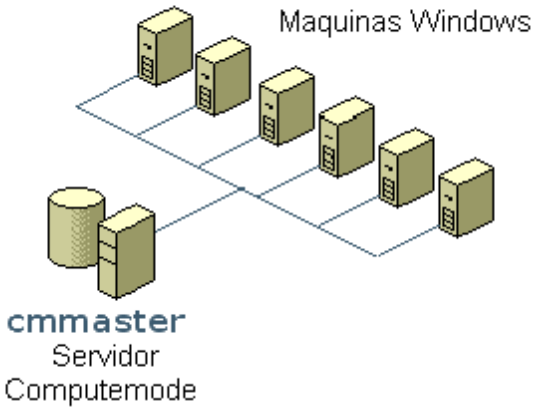

Ilustración 5 Arquitectura Computemode

Fuente: http://computemode.imag.fr/old/sallesdepc/admin.php

Descripción:

Es una infraestructura software basada en Debian Linux que permite extender o crear un Cluster HPC agregando recursos de cómputo no usados. La mayoría de los PCs en las grandes compañías y grandes campus universitarios están ociosas durante la noche, los fines de semana o las vacaciones, entrenamiento de personal, etc.

Computemode se basa en una arquitectura maestro-esclavo usando el servidor central como maestro. Dispone de una interfaz de administración web para un manejo fácil, manteniendo la disponibilidad de los computadores registrados en la red de área local.

Se conocen dos estados de los computadores que se utilizan en el sistema:

- **Usermode:** Es cuando la maquina está ejecutando el sistema que habitualmente usan, que en la mayoría de las organizaciones es Windows.
- **Computemode:** es de donde viene su nombre, es cuando se realizan cálculos para el Cluster.

El sistema ofrece un conjunto de servicios que hacen posible que los recursos reciban por red su sistema operativo y monten su sistema de ficheros raíz por medio de la red. Se basa en la implementación de un disco RAM para la configuración preliminar que luego transfiere el control al sistema de archivos raíz que se implementa totalmente en red a través del protocolo NFS, permite tanto la incorporación de computadoras, como máquinas virtuales.

Un usuario puede enviar trabajos de computo al sistema a través del uso de un clásico manejador de colas logeandose en el servidor (a través de ssh por ejemplo) Computemode emplea OAR para realizar el manejo de colas.

El manejador de colas tiene en cuenta lo siguiente:

- Reservar recursos adecuados para los cómputos.
- Planificar la ejecución de los trabajos.

Ventajas:

- Fácil despliegue: la integración dentro de una infraestructura existente es muy fácil, ninguna modificación es necesaria en los computadores. Computemode viene como una solución única de software, puede llevarse a cabo la integración con los mayores

sistemas de manejo de colas como Sun Grid engine, plataforma LSF y PBS.

- Integración transparente para el Investigador: No se tiene que modificar los trabajos, se trabaja con una interfaz justo como cualquier Cluster Beowulf.
- Integración transparente para el dueño de la computadora: Computemode corre cuando el PC esta ocioso (noche, fines de semana) molestando en lo más mínimo al usuario del PC.
- Cuenta con una interfaz de administración que permite la administrar de los recursos para establecer horarios de cómputo.

Desventajas

- Cuenta con poco software para el cómputo paralelo.
- La versión disponible al público, no cuenta con software actualizado.

IMPLEMENTACION	HERRAMIENTAS HPC POR DEFECTO	ADMINISTRACION DE RECURSOS NO DEDICADOS	ALMACENAMIENTO PERMANENTE	MODO DE IMPLEMENTAR EL SISTEMA DE ARCHIVOS RAIZ
PELICAN	GROMACS, MPI, PVM, OCTAVE, GLANGLIA, SLURM	NINGUNA	NO	SQUASHFS -NFS
DRBL	NINGUNA	NINGUNA	SI	NFS
ONESIS	NINGUNA	NINGUNA	SI	RAMDISK-NFS
COMPUTEMODE	LAM MPI, OAR	INTERFAZ BASADA EN WEB	SI	NFS

Tabla 1 Comparativa de las Implementaciones Diskless

Se escogió Computemode porque es la única solución que cuenta con un completo conjunto de herramientas que permite la fácil administración de recursos tanto a nivel de incorporación y su manejo en horarios definidos, como a nivel de monitorización y la planificación de recursos en un cluster.

4.3.3 DESCRIPCION COMPUTEMODE

La herramienta Computemode se compone de 3 partes principales:

- Interface de Administración de recursos.
- Herramienta de manejo de colas y monitorización de recursos.
- Imágenes de sistema operativo y servicios para el despliegue.

A continuación se detallan cada uno de los componentes.

Interface de Administración de recursos:

Esta interface permite manejar los recursos a utilizar, se basa en una interface web de fácil manejo que interactúa con una base de datos. Aquí se deciden el nombre del nodo, el horario en que será utilizado y la imagen que se desplegara en el. Se puede definir un horario para varios recursos, también permite agregar y eliminar nodos. El procedimiento para agregar nodos permite hacerlo manual o automáticamente lo cual lo hace fácil de administrar.

Herramientas de manejo de colas y monitorización de recursos:

Para el manejo de colas se cuenta con OAR (*Resource Management System for High Performance Computing*) [21], este es un administrador de recursos de computo, el cual es flexible y simple, maneja recursos en un cluser, permite la ejecución de trabajos interactivos y de reserva. En su forma más simple un trabajo se define por un programa a ejecutar, el número de nodos a utilizar y el tiempo por el cual se necesitan los recursos. El trabajo interactivo permite que el usuario le sean asignados los recursos inmediatamente y este interactué directamente con ellos. En un trabajo de reserva, el usuario reserva los recursos para utilizarlos a determinada fecha por un espacio de tiempo definido, y es el usuario el que se encarga de conectarse al trabajo. Para monitorizar los recursos se cuenta con Monika el cual permite visualizar el estado de los recursos mediante una interfaz web, esta se comunica constantemente con el servidor OAR.

Imágenes de sistema operativo y servicios para el despliegue:

Los sistemas operativos que se despliegan en las computadoras se dividen en 2 partes: el kernel Linux y el sistema de ficheros. El kernel

Linux es transferido al nodo de cómputo mientras que el sistema de ficheros es compartido por medio de NFS. Los servicios para realizar el despliegue de los sistemas son los siguientes:
TFTP, DHCP, NFS.

El servicio TFTP cuenta con una serie de scripts que permiten la interacción con la base de datos para la asignación de los diferentes parámetros escogidos por medio de la interfaz de administración. Otros servicios utilizados son: NIS el cual maneja las cuentas de usuario y el servidor DNS el cual permite que todos los nodos se puedan comunicar por medio de nombres.

En el servidor se encuentra la siguiente estructura de directorios importantes para el despliegue, todos estos están bajo el directorio /var/lib/tfptboot/

bin: scripts que consultan la base datos con el fin de proporcionar un archivo al cargador de arranque pxelinux.

etc : Archivos de configuración que son utilizados por el servidor TFTP para permitir la redirección de las peticiones TFTP.

PXEClient: aquí se encuentran el kernel y el ramdisk que se transfieren al equipo cliente, un binario llamado pxelinux.0 que es el cargador de arranque y su respectivo directorio de configuración llamado pxelinux.cfg. Una mirada al directorio se muestra a continuación:

Computemode:/var/lib/tftpboot/PXEClient# ls

APMOFF.CBT	DebianRAM-v1-3.img	linux.0
pxelinux.0	bzImageethersel311.c32	mboot311.c32
pxelinux.0-orig	chain311.c32	fdos1440.img
memdisk	pxelinux311.0	DebianRAM
kerlenny	menu311.c32	pxelinux.cfg

En el directorio pxelinux.cfg se encuentran todos los archivos de configuración de la máquinas, estos archivos pueden tener dos nombres, 01 seguido de la MAC del equipo o default. En el proceso de arranque, el cargador pxelinux buscara un archivo con la MAC del respectivo equipo y si este no se encuentra buscara el archivo llamado default. Ya que Computemode es un ambiente configurable lo cual no

permite que se pueda utilizar un archivo default para todas las máquinas, provoca que se necesite un archivo de configuración por cada máquina agregada al sistema. Para solucionar esto Computemode utiliza los script antes nombrados, los cuales permiten re direccionar la petición hecha por el cargador de arranque y generar un archivo virtual con los parámetros específicos para cada máquina. En este archivo se especifican parámetros como: hostname, dirección maestro, dirección servidor NFS, entre otros. Estos son consultados a la base de datos de Computemode.

La funcionalidad más importante que agrega el mecanismo de generación automática de archivos, es la posibilidad de mantener el servidor funcionando en cualquier momento sin obstaculizar las actividades que las máquinas tienen por defecto, como trabajar con un sistema Windows. Una máquina integrada al sistema Computemode debe configurarse para establecer la tarjeta de red como primer dispositivo de inicio. Cuando una maquina inicia por red, pregunta al servidor si es tiempo de unirse al sistema de computo, si se ha programado así, el servidor le envía un archivo para que inicie por la imagen del sistema de computo; de lo contrario envía otro archivo el cual le indica que inicie por el sistema local, que en este caso será Windows.

En el servidor se emplean dos interfaces de red virtuales, una de estas interfaces es para la comunicación del servidor con el exterior y es la que normalmente utilizan los equipos que se encuentran conectados a la red. La otra de las interfaces se utiliza para la comunicación con los nodos de cómputo, las direcciones IP utilizadas en esta interfaz son privadas y completamente diferentes a las utilizadas por otros equipos. Lo anterior brinda dos ventajas, la primera relacionada con la imposibilidad que tiene un usuario de conectarse a estos equipos desde otra red, en otras palabras estos equipos quedan aislados de las redes externas existentes y la otra ventaja tiene que ver con la no creación de conflictos de direcciones IP provocadas por la utilización de una IP que ya se halla asignado a otro equipo.

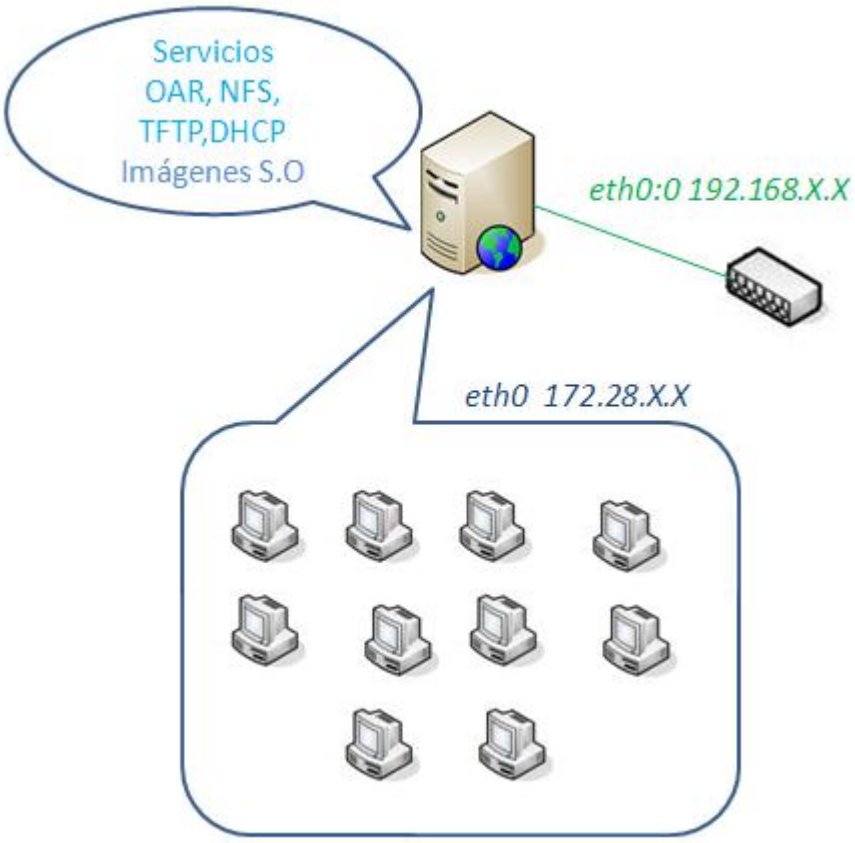

Ilustración 6 Arquitectura detallada de Computemode

5. ARQUITECTURA PROPUESTA

El presente trabajo consistió en la implementación de un prototipo de una infraestructura que permitiera utilizar los recursos ociosos presentes en las salas de informática del campus universitario. Este cuenta con gran cantidad de recursos de cómputo de última tecnología, con características a nivel de hardware que hacen deseable la integración de estos recursos a algún tipo de infraestructura de cálculo principalmente utilizando su tiempo ocioso el cual se pude clasificar en dos tipos:

Tiempo ocioso por fragmentos: Este tiempo ocioso se presenta cuando el usuario deja de utilizar su computadora durante determinado tiempo, normalmente los sistemas operativos ejecutan lo que se conoce como protector de pantalla. Esta clase de tiempo ocioso se caracteriza por su aleatoriedad.

Tiempo ocioso programado: Este ocurre en los horarios nocturnos, o de vacaciones en los cuales normalmente los recursos permanecen apagados.

La implementación de la infraestructura propuesta en este proyecto pretende explotar los tiempos ociosos programados

Limitaciones de Computemode

- La versión que está disponible al público se encuentra desactualizada en lo que se refiere a su núcleo Linux y todos los servicios, esto impediría la instalación de nuevos paquetes software que se necesiten en el sistema.

- Todos los recursos deben estar en la misma red.

- El sistema de archivos es compartido completamente con todos los nodos, lo que genera cierta sobrecarga en la red.

- Solo cuenta con en el entorno de programación paralela LAM/MPI, no dispone de ningún otro software para el computo paralelo o distribuido.

5.1 DISEÑO

La arquitectura propuesta pretende resolver las limitaciones antes mencionadas, además de adaptarse a la arquitectura de red presente en la universidad, de acuerdo a los objetivos planteados. La arquitectura se presenta en la siguiente grafica

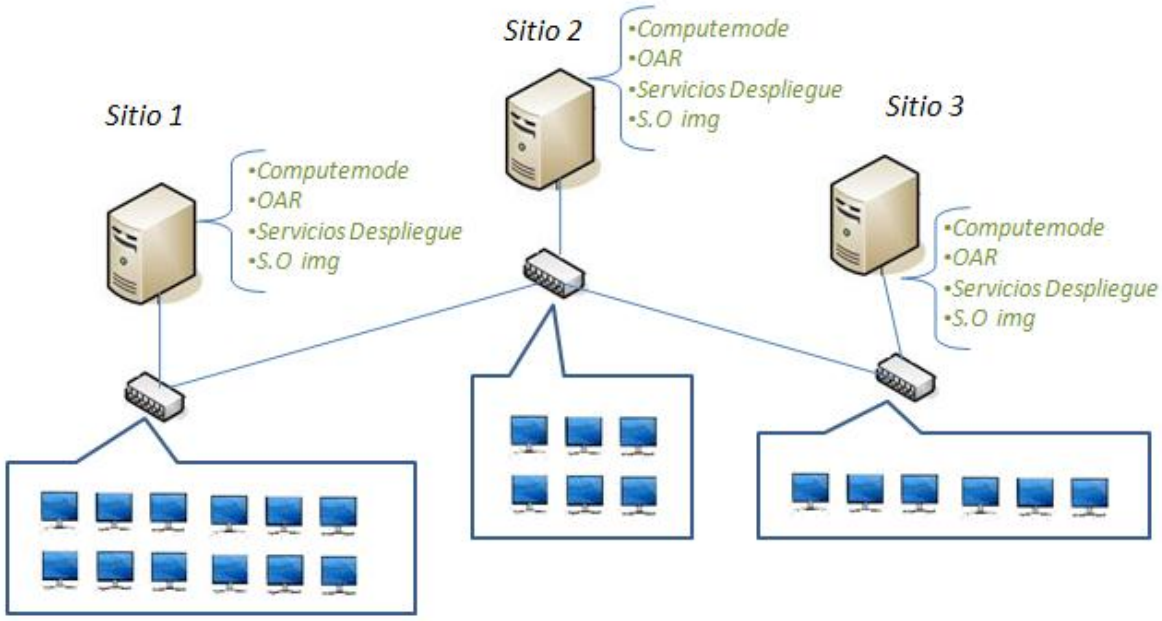

Ilustración 7 Arquitectura General

Principalmente consiste en tener un servidor en diferentes redes de la universidad, cada servidor que será dedicado contara con un conjunto de servicios de administración y configuración de nodos de computo proveídos por Computemode, además de los servicios necesarios para el despliegue y monitoreo, todos estos recursos tendrán una administración descentralizada. Cada servidor contara con un conjunto de herramientas que permitirán la interacción entre los diferentes servidores para proveer un servicio de cómputo distribuido. La propuesta se divide principalmente en dos partes:

Modificaciones a Computemode: Aquí se presenta las modificaciones que se realizaran a nivel de la imagen del sistema Linux a desplegar,

la incorporación de un mecanismo de despliegue de aplicaciones y la modificación de algunos servicios de despliegue.

Arquitectura distribuida: Se presentan los elementos que permitirán la interacción entre los diferentes servidores implementados en cada uno de los sitios.

5.2 MODIFICACIONES A COMPUTEMODE

5.2.1 SISTEMA LINUX PARA LOS NODOS DE CÓMPUTO

Actualmente las implementaciones cluster basadas en diskless presentan ciertos inconvenientes, principalmente porque el sistema de archivos raíz de los nodos clientes es cargado por medio un sistema de archivos de red, generando cierta latencia en la ejecución del sistema como se muestra en [4] y tráfico en la red [5] que aumenta con el número de nodos. Dado que una aplicación paralela hace uso intensivo de la red, provoca que infraestructuras de red que no presentan gran ancho de banda como pueden ser fast ethernet, no sean una muy buena alternativa para el montaje de un cluster diskless ya que degradaría seriamente el desempeño de este.

Teniendo en cuenta lo anterior se buscó una manera de tener un sistema Linux completamente almacenado en RAM, por lo cual el tamaño total del sistema debería ser pequeño con el fin de disponer de suficiente memoria, para la ejecución del sistema y sus respectivos procesos; reduciendo a su vez el tiempo de despliegue de dicho sistema. Lo anterior permitirá obtener un sistema liviano tanto a nivel de espacio requerido en el almacenamiento como en el tiempo de carga del sistema.

5.2.2 SISTEMA DE ARCHIVOS

Con el fin de implementar el sistema de archivos en RAM, se escogió el sistema de archivos *ramfs*, ya que viene integrado con el initramfs, además de que provee un almacenamiento dinámico que puede crecer, lo cual es necesario si se quiere integrar software en caliente.

Otra de las características interesantes es que al implementarlo junto con el ramdisk , el almacenamiento es comprimido en un formato llamando *new c* el cual permite ahorrar cerca del 30 % de espacio almacenamiento en comparación a un sistema de archivos como *ext 3,* ahorrando espacio en RAM que es uno de los propósitos de la implementación.

5.2.3 DESCRIPCIÓN DE LAS IMÁGENES

Las imágenes se componen de una versión liviana del kernel el cual se adapte a las necesidades del ambiente donde se va a ejecutar, que se caracteriza por la no necesidad de manejar cierto hardware. El sistema de Archivos base se almacena en un RAMDISK, este es un sistema Linux mínimo basado en la distribución debían el cual es integrado con un conjunto de herramientas que se describen a continuación:

Ambiente de programación Perl: conjunto de librerías y binarios necesarios para ejecutar los scripts desarrollados en perl, además de contar con librerías que extiende las capacidades del lenguaje, incluyendo comunicación con bases de datos. Este ambiente de programación perl es requisito para ejecutar el Cliente OAR que a continuación se describe.

Cliente OAR: Compuesto por una serie de scripts que permiten la comunicación con el servidor con el fin de establecer algunas variables de entorno que caracterizan a un trabajo y un servidor open-ssh configurado para permitir de manera segura la ejecución de comandos remotos por medio del usuario oar.

Componentes COMPUTEMODE: Conjunto de scripts que se comunican con la base de datos de Computemode ubicada en el servidor, con el fin de determinar el estado de la computadora, si debe reiniciarse o seguir en modo computo de acuerdo con lo establecido por el administrador del sistema. Además permite reiniciar la máquina si el usuario dueño de la computadora necesita de ella eventualmente, se notifica al servidor y esta máquina no estará más disponible hasta que el usuario dueño de la máquina lo quiera así.

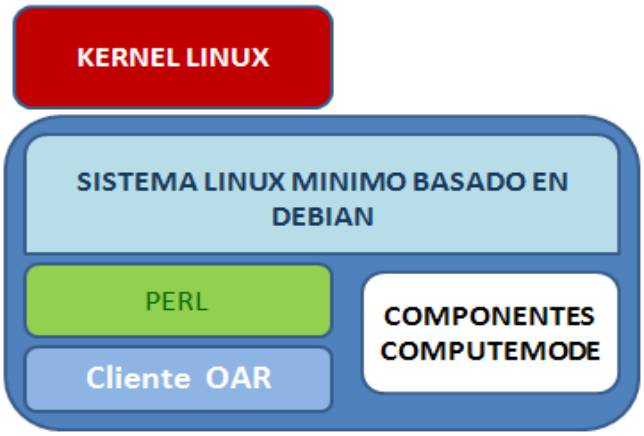

Ilustración 8 Descripción del sistema Linux

5.2.4 PROCESO DE CARGA DEL SISTEMA IMPLEMENTADO

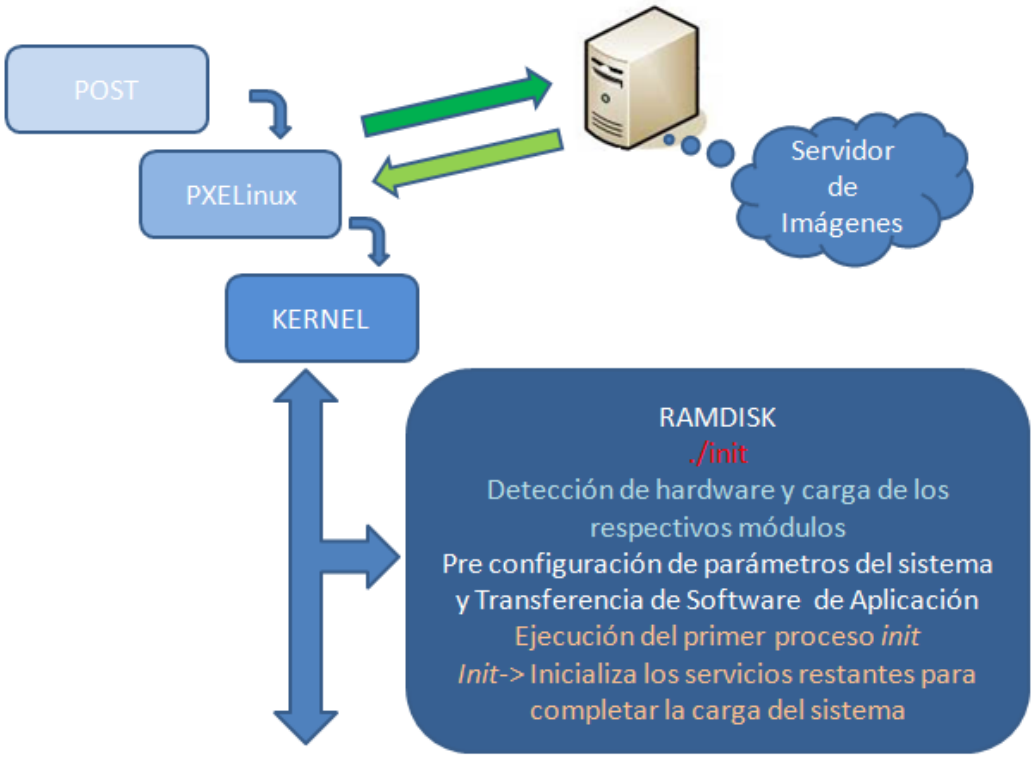

Ilustración 9 Proceso de carga del sistema implementado

El mecanismo de carga implementado se asemeja al de Computemode, utiliza los mismos servicios para el despliegue, la principal diferencia es el uso del RAMDISK como sistema de archivos principal, por tanto no hay necesidad de montarlo por medio de NFS.

5.3 INCORPORACION DE APLICACIONES A LAS IMÁGENES

Objetivo: Integrar diferentes aplicaciones a las imágenes implementadas, esta integración debe realizarse dinámicamente y a petición del usuario lo que permitiría ambientes configurables para diferentes necesidades.

5.3.1 REQUISITOS DEL SISTEMA

- Permitir que todos los usuarios del sistema puedan desplegar las aplicaciones.
- Las aplicaciones deben estar disponibles para integrarse en caliente.
- Permitir a las aplicaciones desplegarse al inicio de cada sistema.
- Instalación automática en todos los nodos que el usuario reservo, no permitir que se pueda instalar en nodos no reservados.
- Integración con OAR.

5.3.2 DESCRIPCIÓN DE UNA APLICACIÓN

Una aplicación en un entorno Linux está constituida por los siguientes componentes:

Binarios: los programas ejecutables, aquí se encuentran tanto los programas propios de la aplicación, como utilidades de las que depende.

Librerías: son un conjunto de funciones comunes para los programas, contiene librerías tanto de las aplicaciones como librerías de dependencia.

Archivos de configuración: archivos que definen el comportamiento de la aplicación.

Además de esto las aplicaciones necesitan que se establezca un ambiente, lo cual consiste en que todos los binarios sean accesibles, sin necesidad de conocerse la ruta donde residen, lo que significa que estén en el PATH del sistema, también se necesitan declarar algunas variables de entorno, etc.

Cada aplicación estará compuesta por su respectivo directorio y un script que permitirá establecer el ambiente de la aplicación. Las aplicaciones estarán asociadas a cada uno de los nodos de cómputo registrados en el sistema. A continuación se presenta las tablas en la base de datos concernientes a las aplicaciones.

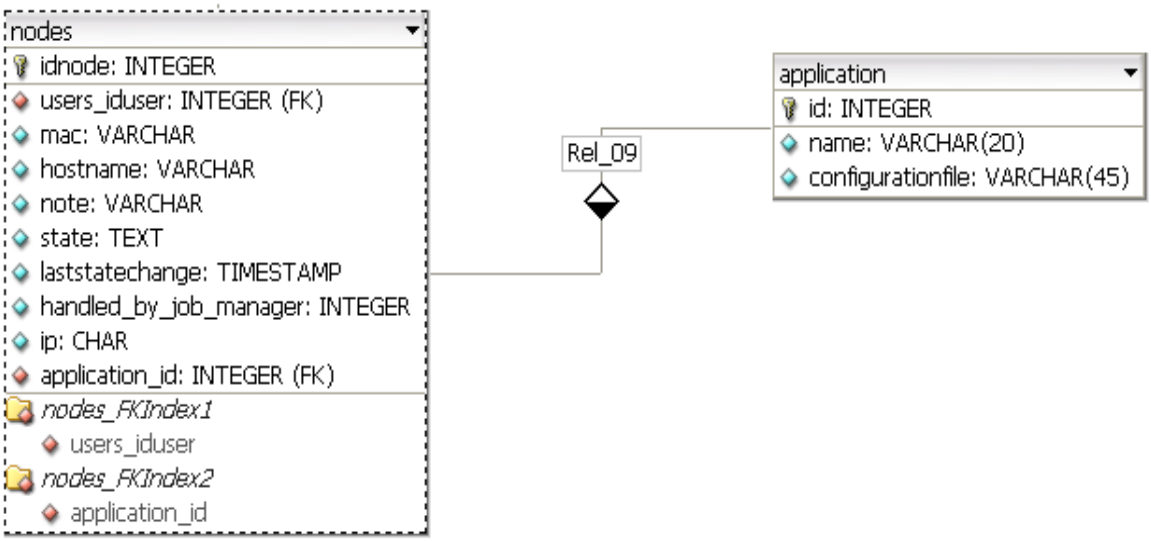

Ilustración 10 Tablas aplicaciones en la base de datos

La aplicación de despliegue se divide en dos partes, la parte del cliente y la parte del servidor

Componente cliente: se encarga de recibir la petición de transferencia de la aplicación proveniente del servidor, transferir la aplicación, configurarla y establecer su ambiente para su correcto funcionamiento.

Componente servidor: se encarga de determinar los recursos reservados, validar los recursos, comunicarse con la parte cliente para que inicien la transferencia de la aplicación.

5.3.3 *INTERACCIÓN DEL USUARIO CON EL SISTEMA DE DESPLIEGUE DE APLICACIONES.*

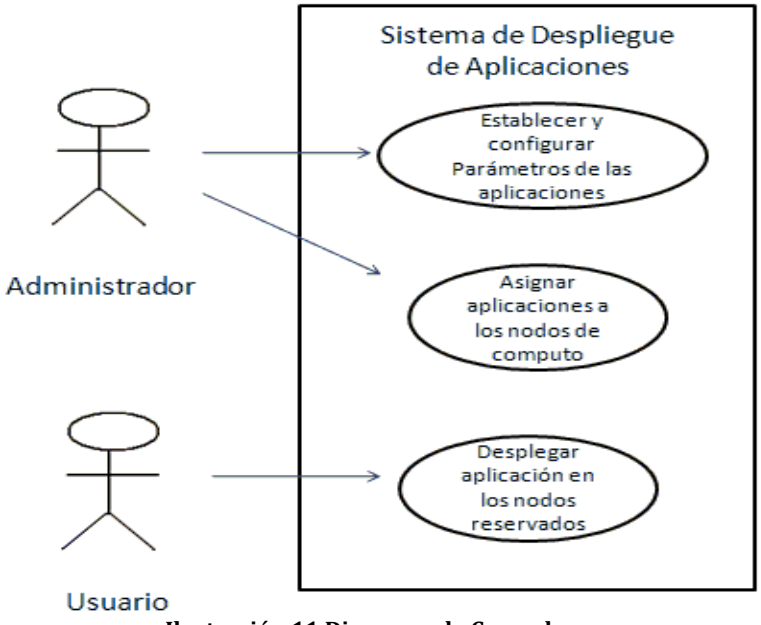

Ilustración 11 Diagrama de Casos de uso

El diagrama de secuencia presenta la interacción del usuario cuando hace una reservación de recursos por medio de OAR y despliega una aplicación específica sobre estos.

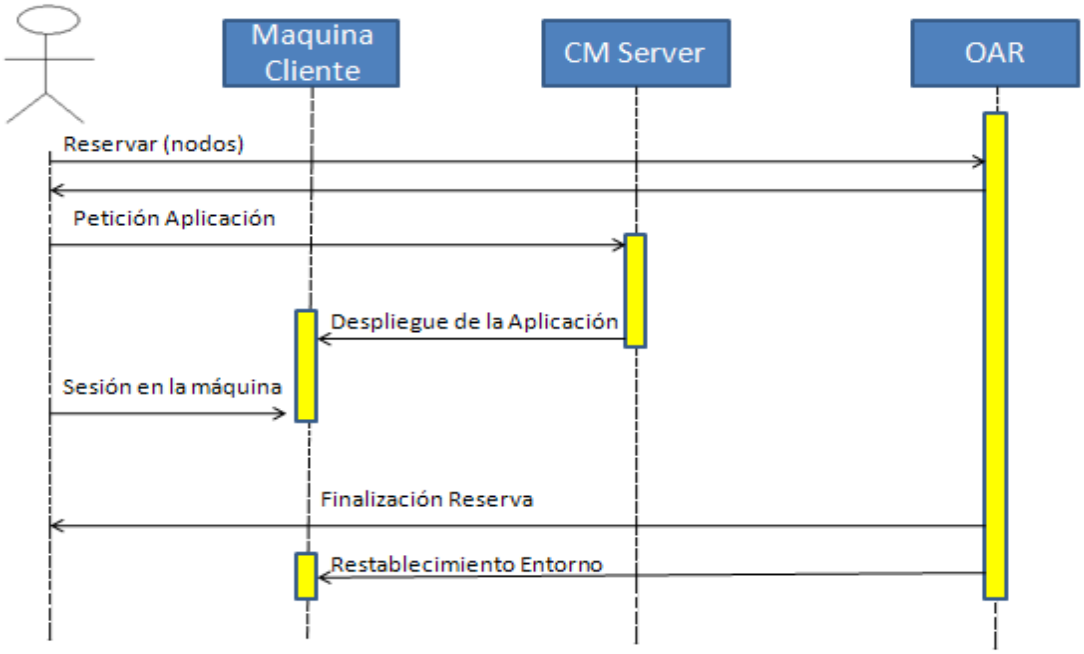

Ilustración 12 Diagrama de Secuencia despliegue de aplicaciones

5.3.4 DESCRIPCION DE LOS COMPONENTES DEL SISTEMA

Los componentes se dividirán en dos clases, los componentes que interactúan directamente con el usuario y los componentes que no tienen interacción directa con él.

Componentes que interactúan con el usuario

Modulo de administración de aplicaciones: Permite Agregar, Editar y Eliminar una aplicación en el sistema.

Modulo de administración de aplicaciones en los nodos: permite asignar, actualizar, eliminar aplicaciones asignadas a los nodos de cómputo, además de mostrar un listado de las aplicaciones asignadas a los nodos.

Modulo de despliegue de aplicaciones: es el que permite el despliegue de la aplicación en los nodos reservados.

Componentes que no interactúan con el usuario

Modulo de Consulta para aplicaciones de despliegue al momento del arranque: Este modulo recibe las peticiones del modulo cliente, realiza las respectivas consultas requeridas y envía información al módulo cliente.

Modulo de integración con OAR: Permite restablecer el entorno de los recursos reservados, una vez la reservación termina.

Modulo Cliente para despliegue de aplicaciones en caliente: permite la ejecución de los respectivos scripts de instalación.

Modulo Cliente para despliegue de aplicaciones en el momento de arranque: permite la ejecución de los respectivos scripts de instalación al momento del arranque.

Ubicación de los distintos módulos

Servidor

Modulo de administración de aplicaciones.
Modulo de administración de aplicaciones en los nodos.
Modulo de despliegue de aplicaciones.
Modulo de integración con OAR.
Modulo de Consulta para aplicaciones de despliegue al momento de arranque.

Nodos de cómputo

Modulo Cliente para despliegue de aplicaciones en caliente.
Modulo Cliente para despliegue de aplicaciones en el momento de arranque.

5.4 ARQUITECTURA DISTRIBUIDA

Se quiere superar la limitación de tener recursos en una sola red, instalando la infraestructura descrita anteriormente en diferentes sitios de la universidad, para permitir que haya un ambiente distribuido que

pueda ser utilizado por los diferentes usuarios, esta arquitectura impone unos retos que se describen a continuación.

5.4.1 DESAFIOS DE UNA ARQUITECTURA DISTRIBUIDA

Comunicación entre los nodos de diferentes sitios: Debido a que los nodos de computo comparten una red privada con el servidor, hace que esta sea inaccesible por fuera de la red y para nodos distintos del servidor; ya que el router que interconecta las diferentes redes no conoce este tipo de direcciones, ocasionando que no sea posible la comunicación entre nodos de diferentes sitios, lo cual no permite la ejecución de un trabajo en forma distribuida.

Almacenamiento: normalmente para aplicaciones que corren en un cluster se cuenta con un tipo de almacenamiento compartido entre todos los nodos pertenecientes a un cluster. Con el fin de modificar en lo más mínimo las aplicaciones que corran utilizando recursos de diferentes sitios y reducir la complejidad de implementación, es deseable que estos recursos de sitios diferentes cuenten con un tipo de almacenamiento en común. Para lograr esto simplemente se puede seguir el mismo modelo de almacenamiento que se implementa a nivel de un mismo sitio, pero la incapacidad de comunicación expuesta anteriormente no permite implementar un almacenamiento de este tipo.

Cuentas de usuario: con el fin de que un usuario pueda utilizar los recursos de diferentes sitios, es deseable que en cada servidor exista su cuenta de usuario, esto con propósitos tanto de usabilidad como de seguridad. Por lo tanto a nivel administrativo la creación de una cuenta debe de reproducirse en todos los sitios pertenecientes a la infraestructura.

Con el propósito de superar los retos antes descritos se exploraron varias soluciones que se describen a continuación:

- Configurar todos los nodos con direcciones IP que se manejen en la red y sean conocidas y enrutables por los routers, esta opción aunque es la más fácil de implementar, impondría problemas de seguridad y posibles conflictos IP.

- Utilizar dos interfaces de red en cada servidor, esta solución posibilita el uso de direcciones privadas y la comunicación entre nodos a través del ruteo de paquetes por las diferentes interfaces. Esto se logra utilizando mecanismos proveídos por IPTABLES que permiten el enmascaramiento IP. Teniendo en

cuenta que esta solución implica modificar el hardware de los equipos existentes, añadiendo otra tarjeta de red a los equipos que vayan a actuar como servidor, lo cual atentaría contra la usabilidad de infraestructuras ya existentes.

- Realizar una configuración dinámica en los nodos que serán utilizados en el computo, a través de la asignación temporal de direcciones IP conocidas por los routers, esto permitirá la comunicación entre los nodos solo en el tiempo en que se realiza el computo, disminuyendo los problemas relacionados con la seguridad y también los posibles conflictos IP, ya que solo se utilizaran un conjunto pequeño de IPs, temporalmente.

5.4.2 PROPUESTA

Se desarrolló un mecanismo que permite ejecutar un trabajo con nodos ubicados en diferentes sitios, este funciona de la diferente manera:

1. El usuario especifica el número de recursos en cada sitio que quiere utilizar.

2. El programa se encarga de consultar a los servidores OAR de cada sitio y de realizar la reservación en cada uno de los sitios a la misma hora y por el tiempo especificado por el usuario. Si el trabajo es un trabajo para correr en paralelo, se crea una llave que se utiliza para la comunicación por ssh.

3. Se crea el ambiente para la comunicación, esto consiste en establecer la IP de la red en cada uno de los nodos, modificar los respectivos archivos para permitir la comunicación por medio de nombres y por ultimo establecer un directorio compartido entre los nodos reservados. El servidor que proveerá este directorio compartido estará ubicado en la misma red en donde se encuentra el usuario que lanza el trabajo.

4. Logear al usuario en un recurso, para que pueda ejecutar su trabajo.

Este mecanismo se divide en dos módulos:

Módulo para la reserva de los recursos.

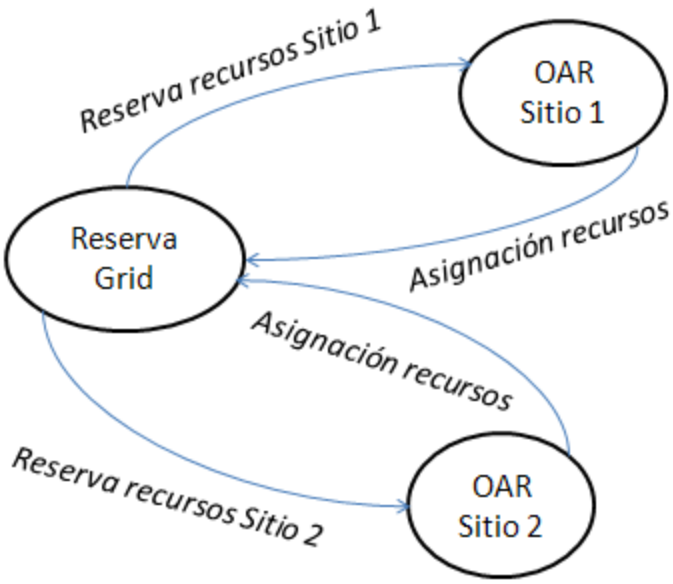

Ilustración 13 Descripción Modulo para la reserva de recursos

Este módulo es el encargado de realizar la reserva de los recursos en los diferentes sitios, interactuando con cada uno de los servidores OAR. En caso de ser necesario el uso de una llave para correr un trabajo en paralelo, se crea la respectiva llave para que pueda interactuar con todos los nodos de forma segura.

Módulo para el establecimiento del ambiente y conexión del usuario al trabajo.

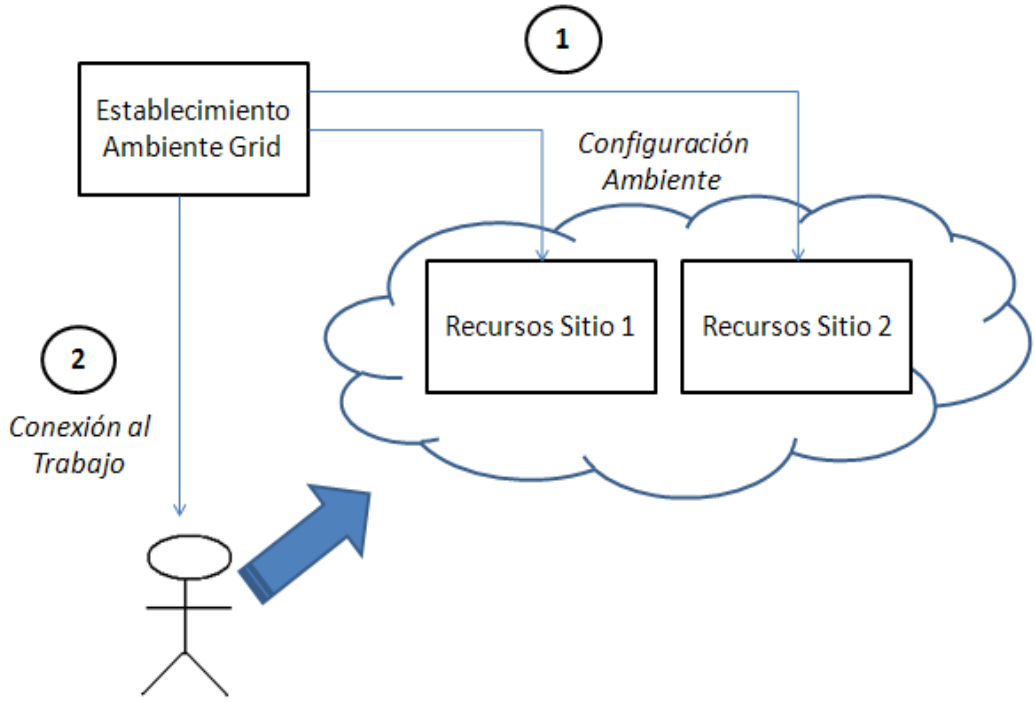

Ilustración 14 Modulo para el establecimiento del ambiente y conexión del usuario al trabajo

Este modulo es el encargado de configurar el ambiente, lo cual consiste en establecer IP de red en cada uno de los nodos reservados por el anterior modulo, además de configurar la resolución de nombres entre ellos y el almacenamiento compartido.

5.4.3 AUTENTICACIÓN

El mecanismo de creación de cuentas de usuario debe tener la capacidad de reproducir esta información por todos los sitios implementados, para tener un ambiente en el cual los usuarios pertenecientes a la infraestructura puedan autenticarse en cualquier sitio. Cualquier cambio en los usuarios tiene que ser reproducido en todos los sitios.

Aunque hay servicios que permiten distribuir el proceso de autenticación en varios servidores, este mecanismo, siempre tiene una arquitectura cliente esclavo, limitando la realización de cambios solo al servidor maestro

6. IMPLEMENTACIÓN Y RESULTADOS

6.1 IMPLEMENTACIÓN SISTEMA LINUX CLIENTE

El almacenamiento del sistema de archivos raíz se hace por medio de *ramfs*, esta tecnología viene integrada con los *ramdisk*. Los pasos para conseguir un sistema almacenado en el *ramdisk* son los siguientes:

- Se creó la estructura de ficheros en un directorio de Linux.
- Convertir este directorio en un *ramdisk*, esta tiene que ser comprimida, para conseguir esto se ejecuta el siguiente comando. (Asumimos que el directorio donde reside la estructura de ficheros se llama OSCliente).

```
cd /OSCliente
find . | cpio --quiet -H newc -o | gzip -9 -n >
/boot/imagefile.img
```

Una vez compilada y comprimida se coloca en el respectivo directorio, junto con el kernel para que sean desplegados a las máquinas cliente.

6.1.1 DESCRIPCIÓN DE LOS SERVICIOS IMPLEMENTADOS EN LAS IMÁGENES

Los sistema Linux cliente solo utilizan el nivel de ejecución 2, este es usado comúnmente para cargar el sistema en modo consola, normalmente un sistema Linux debían mínimo utiliza 35 servicios en el nivel básico y 28 en el nivel 2 para un total 63; a través de la eliminación de servicios innecesarios como chequeo de disco, montaje de swap, limpieza y chequeo de sistema de archivos, servicios de sonido y gráficos, se obtuvo: 23 Servicios en el nivel básico y 10 en el nivel 2 para un total de 33. Lo anterior permitió además de eliminar gran cantidad de software innecesario reduciendo por tanto el tamaño de la imagen, la disminución del tiempo de carga del sistema lo cual minimiza a su vez el tiempo de despliegue que fue uno de los objetivos al implementar la imagen.

Servicios en el nivel básico (rcS)

glibc: Verifica que la versión de las librerías esenciales de C, sea compatible con la versión del kernel.

hostname: Establece el hostname del sistema leyendo el archivo /etc/hostname.

mountkernfs: Monta los sistema de archivos virtuales importantes para el kernel como: /proc, /sys, /var/run, etc.

udev: Inicia el demonio udevd que es el encargado de detectar el hardware de entrada y salida, leer los módulos necesarios, además de crear los archivos respectivos en el directorio /dev.

Mountdevsubfs: Monta el sistema de archivos necesario para udev bajo el directorio /dev

bootlogd: Inicia el demonio bootlogd el cual almacena los logs que se producen al inicio del sistema.

keymap: Establece el idioma del teclado de la consola.

Hwclock: Establece la hora del sistema.

mtab: Realiza el montaje de otros sistema de archivos importantes como: /dev/,/dev/pts muy importante para el funcionamiento de consolas remotas.

Module-init-tool: Carga los módulos que se especifican en el archivo /etc/modules.

Servicios en el nivel de ejecución 2 (rc2)

syslog: Inicia el demonio syslogd el cual se encarga de escribir en el archivo /var/log/syslog todo la información concerniente a los errores, y la salida de los servicios inicializados.

dbus: servicio que permite la comunicación entre aplicaciones por medio de mensajes.

ssh: Inicializa el demonio sshd que permite abrir cesiones remotas por consola.

nis: Inicia el cliente NIS que permite el manejo de cuentas de usuario.

cron: Servicio que permite ejecutar tareas periódicamente.

oar-node: Inicia el cliente oar el cual consiste en un servidor open-ssh con determinados parámetros definidos, como el puerto de escucha 6667.

stop-bootlogd: Detiene el demonio bootlogd.

6.1.2 COMPILACIÓN DE UN KERNEL LIVIANO

Con el fin de reducir el tiempo de carga se compilo un kernel proveído por la pagina www.kernel.org, el cual no realizara la detección de determinado hardware que no se utilizara ni se dispondrá, además de hacerlo más liviano en la inicialización del sistema. Se elimino la detección de discos, sistemas de archivos innecesarios, dispositivo de sonido, dispositivos de entrada salida especiales etc. En el Anexo se presentan más detalles de este proceso.

6.1.3 PROCESO DE CARGA DEL SISTEMA IMPLEMENTADO

El sistema desplegado se inicia en tres etapas.

Primera etapa

Se transfiere el Kernel y el sistema de archivos (RAMDISK) del servidor a cada uno de los clientes. Se ejecuta el kernel y este hace una inicialización de hardware de bajo nivel todo lo necesario en nivel protegido. Luego monta la RAMDISK como sistema de archivos principal, aquí se ejecuta un script llamado *init* que tiene como función cargar módulos de hardware necesarios, como discos, tarjeta de red, teclado, etc.

Segunda etapa

Constituye el establecimiento de la dirección IP en la interfaz de red, la transferencia de las aplicaciones deseadas, algunos directorios entre ellos el directorio */etc*, luego de esta transferencia se establece el hostname modificando el archivo */etc/hostname*. El hostname y la dirección IP son parámetros que hacen a una computadora única en la red, esto son obtenidos de la línea de comando del kernel, más adelante se detalla este mecanismo.

Tercera etapa

Se desmontan los directorios utilizados, y se procede a ejecutar el primer proceso del sistema llamado *init*, el cual se encarga de inicializar los demás servicios.

6.1.4 ESTRUCTURA DE LOS SISTEMAS EN EL SERVIDOR

En el servidor se encuentran los siguientes componentes, el kernel, el ramdisk del correspondiente sistema, y un directorio que contiene la estructura de directorios de un sistema Linux, este es el encargado de proveerle el directorio *etc* durante el arranque y es donde se ubican las aplicaciones. Este directorio se implementa también con el fin de facilitar la instalación de nuevas aplicaciones en la imagen, como puede ser a través del sistema de paquetería junto con la utilidad *chroot*.

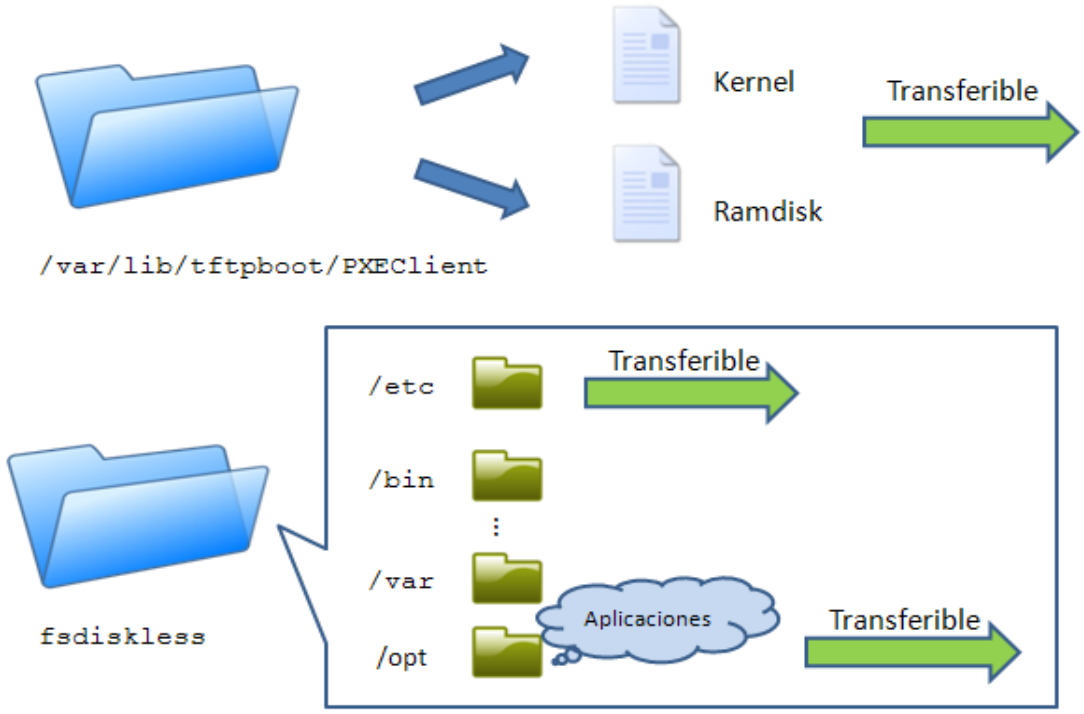

Ilustración 15 Estructura de Directorios en el servidor

6.2 MEJORAS EN LOS MECANISMOS PRESENTES

Aquí se describe las mejoras que se realizaron en dos mecanismos utilizados por la infraestructura, con el objetivo de disminuir el tiempo de despliegue y la complejidad de otros servicios que se cree innecesaria. Los mecanismos en los que se intervino, fueron el mecanismo de asignación IP y el mecanismo de asignación de nombres.

La asignación de las direcciones se realiza mediante el servicio DHCP, este servicio en Computemode se utiliza en dos momentos, al momento de la petición por parte del NIC de la tarjeta y en el proceso de arranque del sistema, dado que este proceso de petición-concesión toma algunos segundos se considero eliminar la asignación de la IP a la interface de red por este método, con el objetivo de disminuir el tiempo de despliegue. El mecanismo utilizado para suplir el servicio DHCP fue la asignación de una IP por medio de la línea de comandos

del kernel. Para lograr esto se modificó la base de datos de Computemode, la cual almacena toda la información concerniente a los nodos de computo, para añadir un campo en la tabla de nodos y guardar la IP de cada nodo, además se creó una función que permitía la asignación de una IP en el momento de registro del nodo en la interfaz de administración.

Computemode utiliza el servicio DNS para la comunicación entre nodos por medio de nombres, tener un servicio de este tipo para un ambiente que no tiene posibilidades de crecer mas allá de cierta cantidad de nodos, no se justifica, en vez de esto se implemento un mecanismo que con base en las modificaciones ocurridas en la interfaz de administración de Computemode, modifica el archivo */etc/hosts* que es el archivo donde primero consulta todo sistema Linux a la hora de resolver un nombre de una máquina. Así se tiene un mecanismo simple que permite la comunicación entre nodos por medio de nombres y que está acorde con el sistema.

6.3 IMPLEMENTACIÓN MECANISMO DE DESPLIEGUE

Para la implementación de los módulos, se manejaron lenguajes como: Perl, bash y php.

6.3.1 MECANISMO DE DESPLIEGUE AL MOMENTO DE ARRANQUE DEL SISTEMA

Este mecanismo consta de dos módulos que se describen a continuación.

Modulo de Consulta para aplicaciones de despliegue al momento de arranque

Este modulo se implemento en php, por facilidad de comunicación a través de la URL. El trabajo de este modulo es realizar la consulta a la base de datos.

El modulo se compone de:

- Archivo *get_appconf.php* recibe por medio de la URL la MAC del equipo cliente, se encarga de verificar que sea una MAC valida, establece cierta información de depuración y ejecuta una función que realiza la consulta a la base de datos e imprime la información respectiva retornada por la función.

- Funcion *db_get_default_application* nombrada anteriormente, recibe como parámetro la MAC del equipo, realiza la consulta y retorna el resultado de la consulta.

Módulo cliente para despliegue de aplicaciones en el momento de arranque

Este modulo consiste en un script, que se comunica con el Modulo de Consulta a través del archivo *get_app.conf.php* utilizando la URL, recibe de este un archivo que es el resultado de la consulta, este archivo trae especificado el script de configuración a ejecutar. Finalmente ejecuta este script.

6.3.2 SCRIPT DE INSTALACION DE APLICACION

```
#/bin/sh

nfsmount -o ro,nolock 172.28.255.253:/fsdiskless /root.new/
echo -n "Installing NAMD ....."
######### NAMD INSTALLATION ######################################
cp -ra  /root.new/opt/namd              /opt/
echo -n  "."
cp -ra  /root.new/opt/NAMD_2.6_Source     /opt/
echo -n  "."
ln -s /opt/NAMD_2.6_Source             /opt/namd/.rootdir
cp -ra /root.new/usr/lib/libtcl8.4.so.0 /usr/lib/
cp -ra /root.new/usr/lib/libsrfftw.so.2.0.5 /usr/lib/
ln -s /usr/lib/libsrfftw.so.2.0.5        /usr/lib/libsrfftw.so.2
cp -ra /root.new/usr/lib/libsfftw.so.2.0.5 /usr/lib/
ln -s /usr/lib/libsfftw.so.2.0.5        /usr/lib/libsfftw.so.2
##########Configuring Environment ##############################
cat /home/$1/.envbasic >/home/$1/.bashrc
echo "export PATH=/opt/namd/:\$PATH"> /home/$1/.bashrc
umount /root.new/
```

Ilustración 16 Ejemplo de script de instalación

En el grafico se muestra un ejemplo de script de instalación, en este caso para la aplicación NAMD, ese script se puede dividir en 3 partes principales, la primera tiene como función montar un directorio desde el servidor en el cual están ubicadas las aplicaciones. En la segunda parte se realiza la transferencia de los diferentes directorios y librerías necesarias, para luego en la última parte configurar el ambiente de la aplicación, declarando en el PATH del sistema la ruta en la cual se encuentra los binarios de la aplicación y desmontando el directorio del servidor.

6.3.3 MECANISMO DE DESPLIEGUE EN CALIENTE

Este mecanismo es el que se encarga de realizar el despliegue de la aplicación a los nodos reservados. A continuación se describe el proceso que se realiza para el despliegue de una aplicación.

1. Consulta a la base de datos, para saber el script de instalación que se utilizara teniendo en cuenta la aplicación elegida por el usuario, se obtiene la ruta completa del script de instalación. Estos scripts de instalación se encuentran en cada máquina cliente.
2. Lee el archivo de máquinas, en el cual se especifican los nodos reservados, este archivo es creado por OAR una vez se crea la reserva. El programa confirma que los nodos especificados en el archivo sean los nodos que se reservaron.
3. Ejecuta remotamente los scripts de instalación en cada una de las máquinas especificadas en el archivo nombrado anteriormente.

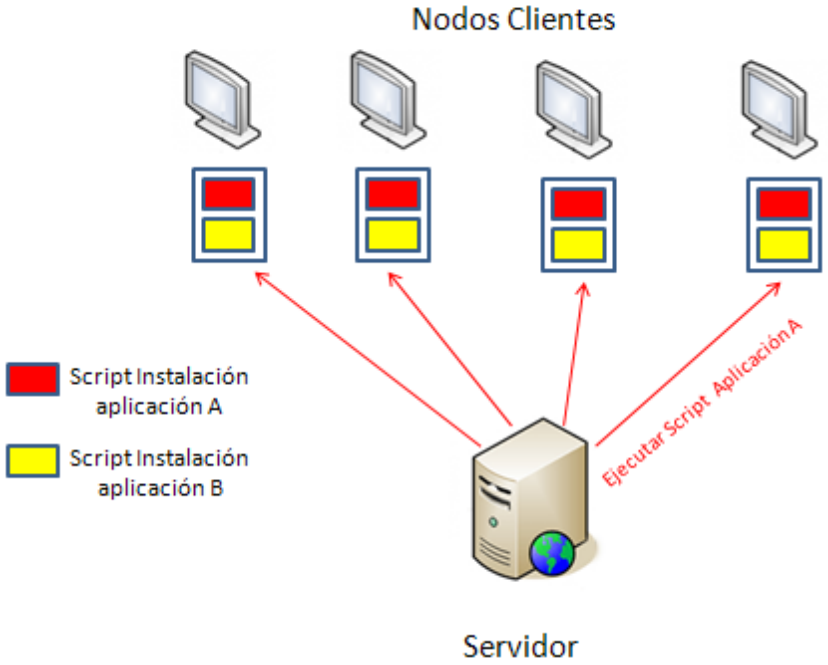

Ilustración 17 Mecanismo de despliegue en caliente

6.3.4 MODO DE TRASNFERENCIA DE LA APLICACIÓN

Para la transferencia de las aplicaciones del servidor a los nodos de cómputo se escogió la transferencia por NFS. Este mecanismo es bastante simple, consiste en montar en las computadoras clientes el directorio donde se encuentra la aplicación y realizar una copia de la aplicación, a un directorio local. Se escogió este tipo de transferencia, por lo que no se necesita tener una aplicación adicional en la computadora cliente, como podría ser un cliente FTP en el caso de que esta transferencia se realizara por FTP; además de ser muy simple de implementar.

Aunque este mecanismo es muy simple, presenta una desventaja, la cual se relaciona con la imposibilidad que tiene un usuario común para realizar el montaje de un directorio por NFS en un directorio del sistema, estas tareas normalmente son hechas por el usuario root. Lo anterior atentaría contra el requisito de que todos los usuarios pudieran desplegar la aplicación. Con el fin de solucionar este inconveniente se uso la utilidad del sistema sudo, la cual permite a través de una

adecuada configuración la ejecución de scripts con los privilegios de determinados usuarios. Con esto se consiguió que todo usuario lograra ejecutar el script (que a continuación se describe) sin perjudicar de ningún modo la seguridad.

6.3.5 INTEGRACION CON OAR

Para desplegar las aplicaciones en un conjunto de nodos, estos deben ser reservados, el tipo de reserva será el interactivo el cual se describe a continuación.

Reserva interactiva en OAR

Cuando un usuario reserva un conjunto de recursos de forma interactiva, el usuario es logueado directamente en uno de los recursos asignados, y el manejador de colas OAR se encarga de establecer ciertos parámetros como: los hostnames de los equipos reservados, el número del trabajo, las propiedades de los recursos etc. La reserva tiene un tiempo Limite, que por defecto es de 2 horas, por lo tanto esta reserva es terminada o este trabajo es terminado por el servidor OAR una vez se haya cumplido el tiempo de la reserva o el usuario abandone los recursos.

Como el despliegue de la aplicación se realiza desde el servidor a los nodos, es necesario ejecutar el mecanismo de despliegue desde el servidor, por lo tanto se debe configurar el servidor OAR o los recursos, para permitir que al realizar un trabajo interactivo este no nos loguee inmediatamente en el primer recurso reservado si no que permanezca en el servidor, esto puede ser logrado mediante la opción deploy, que viene con OAR, esta opción permite que cuando se ejecute un trabajo y si este se especifica con el tipo de trabajo deploy entonces se loguee en el servidor en vez del primer nodo reservado. Para permitir lanzar un trabajo de despliegue se debe configurar los recursos mediante los comandos que provee OAR de la siguiente manera:

oarnodesetting -h nodo1.computemode.lan –-property=”deploy=YES”

Realizamos lo anterior con todos los recursos en los cuales deseemos realizar el despliegue de aplicaciones, una vez configurados, la reserva se hace de la siguiente manera:

oarsub -I -t deploy -l /nodes=3

Aquí se reservan 3 nodos para despliegue, esto nos logueara en el servidor. Una vez en el servidor podemos desplegar la aplicación en los nodos reservados a través de las utilidades implementadas.

Otro aspecto que debe configurarse es el restablecimiento del ambiente en los nodos reservados una vez la reserva ha terminado, OAR cuenta con unos scripts llamados epilogue y prologue que son ejecutados al iniciar y finalizar cada trabajo respectivamente. Por defecto OAR no ejecuta ninguno de ellos, por lo tanto se debe configurar el servidor para que ejecute cierto script al final de cada trabajo, este script recibe como parámetro el numero del trabajo. El restablecimiento del ambiente en cada uno de los nodos consiste simplemente en el reinicio de cada uno de los recursos para permitir que el sistema pueda cargarse nuevamente, ya que no es deseable que el reinicio ocurra siempre sino solo con los trabajos de despliegue, se tuvo que realizar un proceso de verificación del trabajo a través del numero del trabajo y comandos de administración que provee OAR, lo cual permitió diferenciar entre un trabajo normal y uno de despliegue. Luego de la verificación, si el trabajo es de despliegue se procede al reinicio del los recursos a través del comando implementado *appreboot*, que es un script desarrollado en perl que permite el reinicio remoto de los recursos.

6.3.6 APLICACIONES INTEGRADAS

POVRAY

Es un programa que crea imágenes tridimensionales foto-realísticas usando una técnica de renderizado llamada trazado de rayos. Lee de un archivo de texto información que describe el objeto y la iluminación en la escena y genera una imagen de la escena desde el punto de vista de una cámara que también es descrita en el archivo de texto. La técnica de trazado de rayos no es un proceso rápido, pero produce imágenes de gran calidad con reflexiones realistas, sombras, perspectivas y otros efectos.

NAMD

Es un software de dinámica molecular para plataformas UNIX que corre en paralelo diseñado para la simulación de alto desempeño de grandes sistemas biomoleculares. NAMD usa el popular programa de visualización VMD para el establecimiento de la simulación y el análisis de la trayectoria. Las simulaciones de dinámica molecular calculan las trayectorias atómicas resolviendo numéricamente ecuaciones de movimiento usando campos de fuerza empíricos, como el campo de fuerza CHARMM, que aproxima la fuerza real ejercida por los átomos en sistemas de biopolimeros.

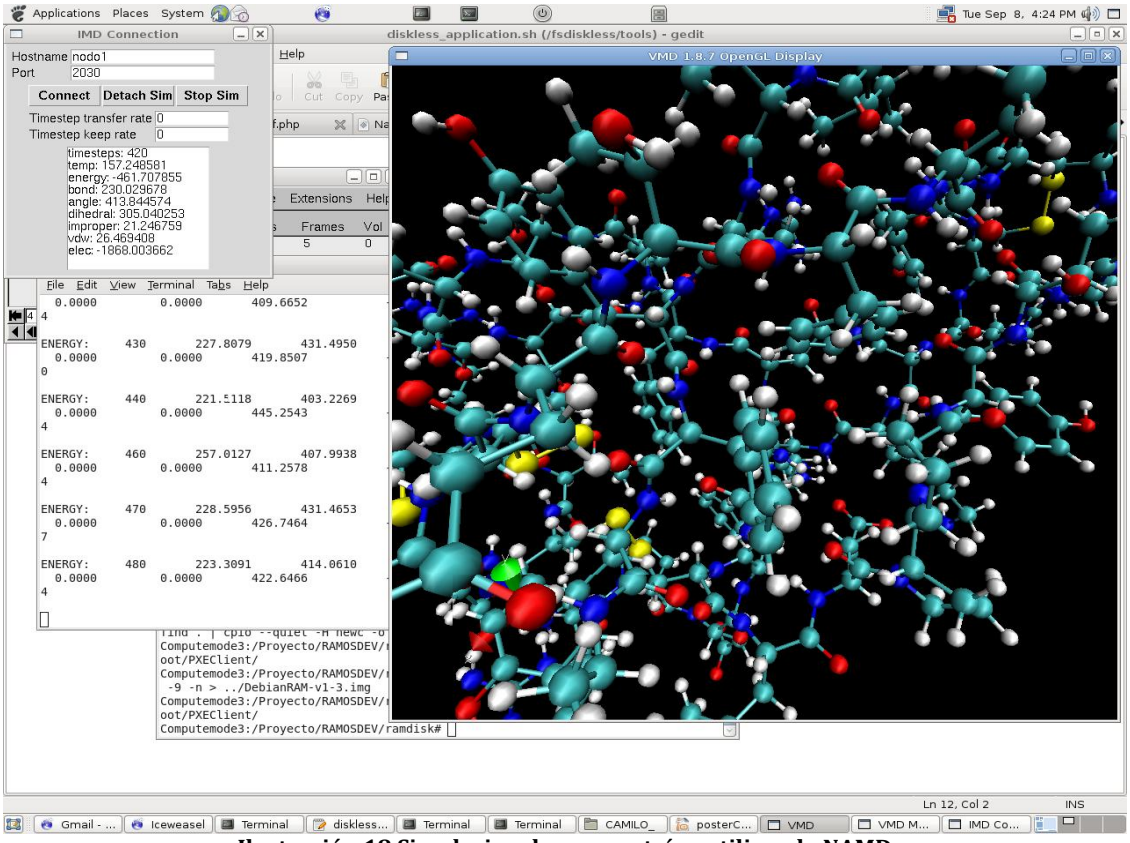

Ilustración 18 Simulacion de una proteína utilizando NAMD

6.4 ARQUITECTURA DISTRIBUIDA

Se implemento la infraestructura en diferentes salas de informática de la universidad, estas salas están ubicadas en: sistemas y el CENTIC,

en cada uno de estas salas se encuentra un servidor Computemode, con todos los servicios como son: OAR, interfaz administrativa, Servicios de despliegue, Imágenes Implementadas y el Mecanismo de despliegue de aplicaciones. Se configuro la infraestructura para que los usuarios de esta pudieran loguearse en cualquier servidor, por tanto sus usuarios deben existir en todos los servidores, así que se creó un mecanismo para la creación de usuarios en cada servidor, que permitiera la reproducción de estos usuarios en cada uno de los servidores. Así los usuarios pueden acceder a cada servidor y lanzar sus trabajos en ellos, interactuando con el respectivo manejador de colas de cada sitio.

6.4.1 AUTENTICACIÓN

Se creo un mecanismo que permite la creación de usuarios y la reproducción de estos en cada uno de los servidores que conforman la infraestructura.

Cuando se crea un nuevo usuario en Linux, se modifican los siguientes archivos: passwd y el shadow, en los cuales se define el nombre del usuario, su carpeta personal, la consola por defecto, su clave encriptada, etc. Además de esto se crea un directorio personal que normalmente reside en home.

El mecanismo consiste en crear el usuario mediante el comando *adduser* en un servidor y modificar los respectivos archivos antes mencionados en los demás servidores de la infraestructura, con lo cual se consigue que el usuario pueda autenticarse en cualquiera de los sitios. El script para la creación de usuario en forma distribuida realiza las siguientes acciones:

1. Crear el respectivo usuario con el comando *adduser*
2. Actualizar la base de datos del servicio NIS
3. Actualizar los respectivos archivos, crear los directorios y actualizar base de datos del servicio NIS, en los otros servidores presentes en la infraestructura.

6.4.2 TRABAJO DISTRIBUIDO

Reserva distribuida

Se reserva recursos en un sitio según lo escogido por el usuario, lo primero es crear una reserva en cada uno de los sitios por el numero de nodos que usuario desee, a continuación se muestra:

```
SITIO1=”cmserver.computemode.local
SITIO2=”cmserver2.sistemas.local
ssh $SITIO1 oarsub -l "nodes=$NODES1" -e my_key -r '$FECHA
$HORA' –t deploy
scp /home/oargrid/mykey SITIO2:/home/oargrid/
ssh $SITIO2 oarsub -l nodes=$NODES2  -i my_key -r '$FECHA
$HORA' –t deploy
```

Una vez hecha la reservación se deben configurar los respectivos nodos, esta configuración consiste en establecer nuevas IP’s en los nodos reservados y configurar un almacenamiento compartido. Para establecer las nuevas IP’s se procede como sigue:

ssh nodo1.compuemode.local eth0 192.168.109.231 up
echo “192.168.109.231 nodo1.computemode.local “>>hosts

El script implementado automatiza esta tarea, a través de la consulta del archivo generado por OAR en el cual se encuentra especificado los nodos reservados, y consulta una IP de un mecanismo que asigna las ultimas IP’s disponibles en el rango de la red para evitar posibles conflictos. Lo anterior se realiza en cada uno de los servidores en los cuales se ha lanzado el trabajo.

Actualizar el archivo host generado automáticamente en los procedimientos en cada uno de los servidores, en todos los nodos pertenecientes en la reserva

Configuración almacenamiento

umont /home/
mount 192.168.109.139:/home/ /home/

En este caso en cada uno de los nodos es desmontado el directorio /home, el cual vuelve a montarse pero desde el servidor que tienen en común. Este servidor pertenece a la misma red en donde el trabajo es lanzado, todos los nodos acceden a este servidor para almacenamiento tanto nodos presentes en la misma red, como nodos presentes en otra red.

Una vez configurados todos los componentes se procede a conectar al usuario al trabajo, por medio del siguiente comando:

oarsub -C IdJob

A continuación se muestra una tabla comparativa de las implementaciones Diskless estudiadas y la implementada en este trabajo, mostrando las características mas relevantes.

IMPLEMENTACION	HERRAMIENTAS HPC POR DEFECTO	ADMINISTRACION DE RECURSOS NO DEDICADOS	ALMACENAMIENTO PERMANENTE	MODO DE IMPLEMENTAR EL SISTEMA DE ARCHIVOS RAIZ
PELICAN	GROMACS, MPI, PVM, OCTAVE, GLANGLIA, SLURM	NINGUNA	NO	SQUASHFS -NFS
DRBL	NINGUNA	NINGUNA	SI	NFS
ONESIS	NINGUNA	NINGUNA	SI	RAMDISK-NFS
COMPUTEMODE	LAM MPI, OAR	INTERFAZ BASADA EN WEB	SI	NFS
NUEVA IMPLEMENTACION	MPICH, OPEN-MPI, OAR, POVRAY, NAMD	INTERFAZ BASADA EN WEB	SI	RAMDISK

Tabla 2 Tabla comparativa de las implementaciones Diskless y la implementación desarrollada

6.5 RESULTADOS

6.5.1 DESCRIPCIÓN DEL AMBIENTE DE PRUEBAS

El ambiente de pruebas consistió en 70 máquinas distribuidas en tres salas con las siguientes características: Intel Pentium 4 CPU 3.2 GHz, 1 GB de memoria RAM, la arquitectura cuenta con un PC como servidor, este PC esta ubicado en una sala que posee una interconexión de red de tipo Gigabit, las otras dos salas cuentan con Fast ethernet.

6.5.2 PRUEBAS DE DESPLIEGUE

La primera prueba consistió en medir el tiempo de despliegue de un determinado número de máquinas simultáneamente. Estas pruebas se realizaron por los siguientes motivos:

- Medir si la implementación de una imagen más liviana y la optimización de servicios redujeron el tiempo de despliegue, con respecto al Computemode.

- Dado que ahora se transfiere más por medio de la red, esto aumenta considerablemente el tráfico de la red, se quiere ver la escalabilidad del sistema con respecto al despliegue, además comprobar que no se sature los servicios de red durante este, dejando al switch inhabilitado provocando la caída de toda la red.

Las pruebas se realizaron en un total de 40 máquinas mediante el siguiente proceso:

1. las máquinas a desplegar, son encendidas por medio de la utilidad *etherwake*, esto permite un inicio casi simultáneo de todas las computadoras cliente.

2. Una vez las máquinas son encendidas por el servidor, este ejecuta un programa que verifica constantemente si se ha escrito un determinado valor en un archivo, este ejecuta una función que mide el tiempo transcurrido hasta que el programa termine su ejecución.

3. Cada máquina cliente cuenta con un script, que es ejecutado por el último servicio en inicializarse. Este script se encarga de aumentar un contador que se escribe en un fichero almacenado en el servidor.

4. El programa ejecutado en el servidor al encontrar el valor determinado, lo que quiere decir que el número de nodos requerido han completado su inicialización, termina su ejecución y muestra el tiempo transcurrido que constituye el tiempo de despliegue.

En la grafica a continuación se muestran los resultados experimentales obtenidos contrastados con Computemode.

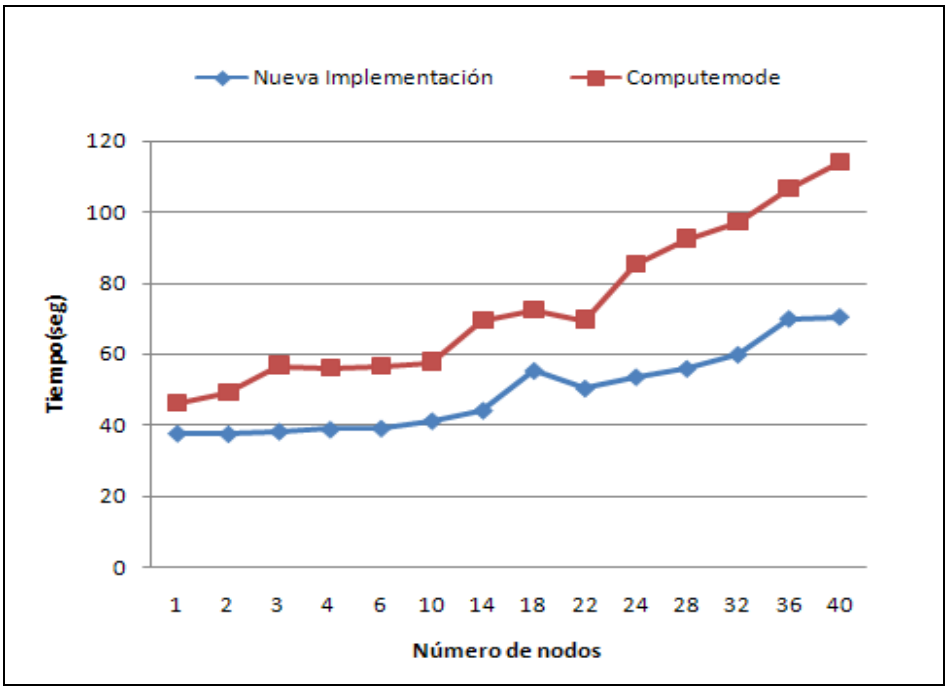

Ilustración 19 Resultados prueba de Despliegue

6.5.3 EJECUCION DEL BECHMARK HPL (HIGH PERFORMANCE LINPACK)

Linpack es un bechmark utilizado para evaluar la capacidad de procesamiento de las supercomputadoras, de uso muy difundido en la comunidad de cómputo de alto rendimiento. Es usado para generar el ranking del Top500[9], el cual muestra las 500 computadoras más rápidas del mundo. Este benchmark no pretende medir el desempeño general de un sistema, en vez de ello, evalúa el desempeño en un área muy específica, el cálculo de sistemas de ecuaciones de alta densidad. Esta medida resulta útil para conocer las capacidades del equipo de computo de alto rendimiento, pues esta clase de equipos generalmente se utilizan para resolver este tipo de problemas, de tal forma que el benchmark linpack proporciona una idea bastante acertada del desempeño que el equipo tendrá en aplicaciones reales.

En general, el benchmark realiza la solución de un sistema de ecuaciones generado aleatoriamente, expresado como una matriz de coeficientes que en la computadora están representados por números en coma flotante, normalmente a una precisión de 64 bits. El paso

[9] Top 500 supercomputer sites, http:/www.top500.org

crucial de esta solución es la descomposición LU con pivoteo parcial de la matriz de coeficientes. Obtener la solución requiere $2/n^3 + 2n^2$ operaciones de punto flotante, donde n es la dimensión de la matriz. Finalmente el benchmark entrega un valor de rendimiento expresado en Gflops (miles de millones de operaciones por segundo). Este valor es el que se suele emplear al efectuar la comparación entre diversos equipos.

LINPACK fue originalmente implementado en lenguaje fortran para máquinas uniprocesador, vectoriales y SMP. A medida que los equipos MPP fueron cobrando auge, se hizo necesario contar con una manera de comparar su desempeño, de forma que se desarrollo el benchmark HPL [20](High Performance LINPACK). HPL es una implementación del benchmark linpack escrita en lenguaje C, que puede ejecutarse en casi cualquier equipo que cuente con una implementación MPI, lo cual incluye prácticamente cualquier equipo MPP comercial y desde luego cluster tipo Beowulf.

El interés que se tuvo al ejecutar esta tipo de prueba, fue medir la capacidad de procesamiento del cluster implementado, así como su escalabilidad. Otro aspecto que permitiría evaluar, era la ejecución de procesos demandantes en memoria en el sistema Linux implementado. Al alcanzar determinado tamaño de problema por ejemplo 17000, esto corresponde a solucionar un sistema de ecuaciones de tamaño 17000, por lo tanto el tamaño de la matriz en lo que se refiere a espacio de almacenamiento es de:

17000 x 17000 x 64 = 18496000000 bits = 1.849 GBs

Lo cual se distribuye en la memoria de todos los nodos. Esta prueba se realizo en 24 equipos de cómputo, los resultados se muestran en la grafica a continuación.

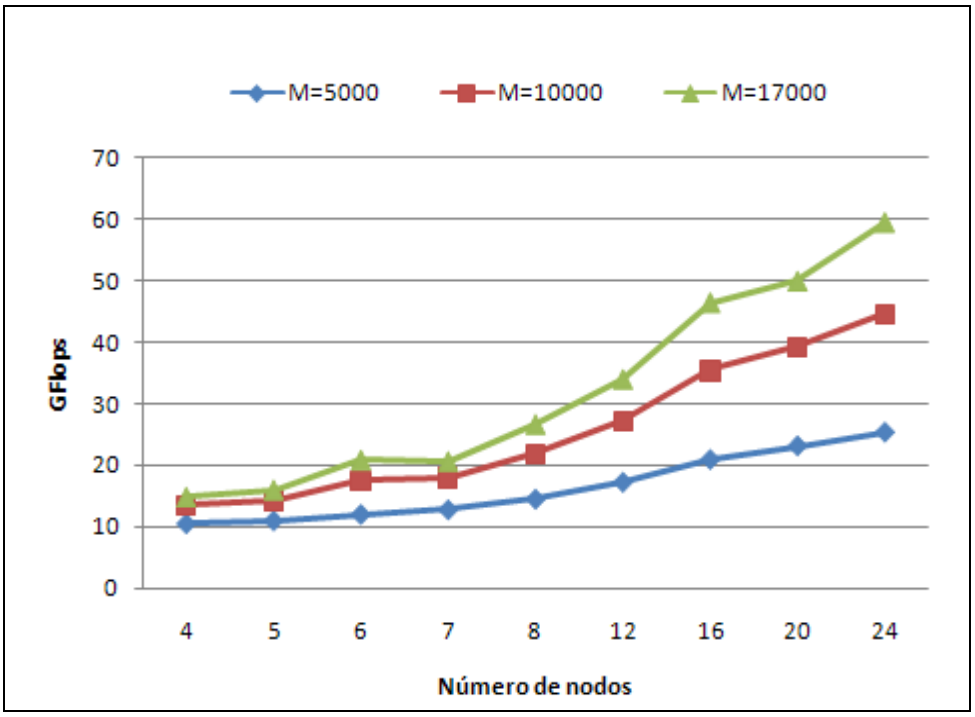

Ilustración 20 Resultados pruebas de rendimiento

Con el fin de medir la capacidad de procesamiento con determinado número de nodos, se realizaron pruebas con diferentes tamaños de sistemas de ecuaciones a solucionar, los cuales están representados en la grafica por la letra M. El máximo rendimiento obtenido fue de 59.49 Gflops que corresponden a 24 nodos.
Este rendimiento puede ser comparado con el rendimiento teórico de una computadora súper escalar con la siguiente ecuación [8]:

$$Rpeak = ncores \cdot nFPU \cdot f$$

En nuestro caso ncores=24, nFPU=3.2 Ghz, f=1 y remplazando obtenemos:

$$Rpeak = 24*3.2*1=76.8\ Gflops$$

Lo anterior nos permite calcular la eficiencia:

$$Eficiencia= (Rendimiento\ Experimental / Rendimiento\ Teórico)*100$$

$$Eficiencia= (59.49 / 76.8)*100 = 77.46\ \%$$

6.5.4 PRUEBA SISTEMA EN DISCO VS SISTEMA EN RAM

Se midió el rendimiento de un sistema cluster con disco, en las mismas condiciones de red y hardware presente en las máquinas, y este se comparo con el rendimiento del software implementado, el rendimiento se midió con el benchmark HPL para un tamaño de problema de 10000, en estas pruebas se utilizaron un máximo de 10 máquinas.

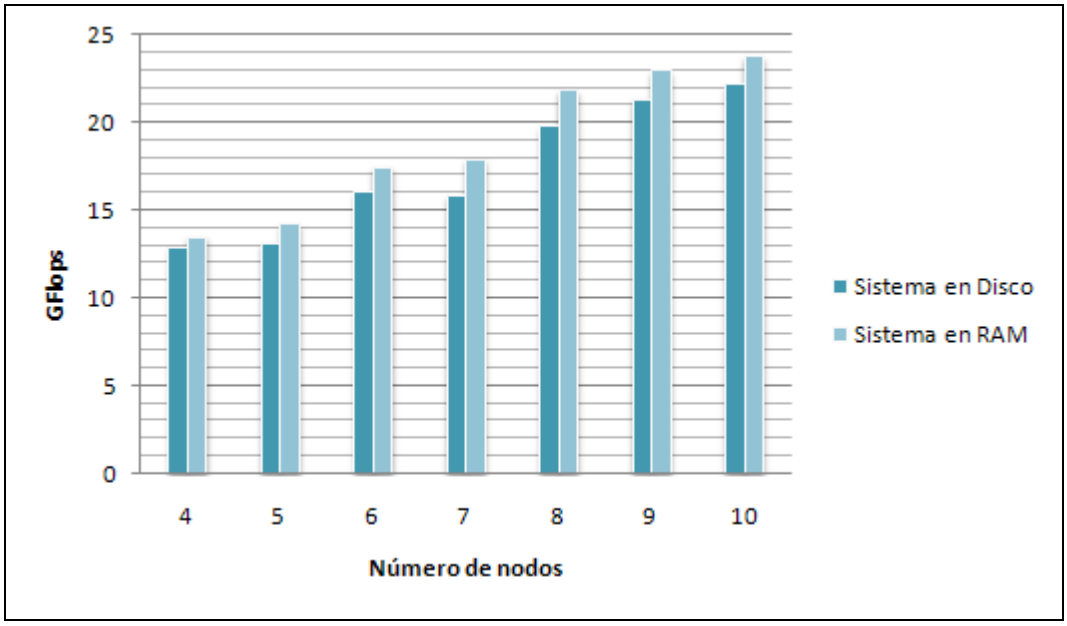

Ilustración 21 Resultados Comparación cluster implementado vs cluster en disco

De la grafica se puede observar que el sistema implementado en RAM, presenta un mejor rendimiento comparado con el sistema que se almacena en Disco, aunque la diferencia no es muy significativa, permite deducir que en el peor de los casos un cluster no dedicado, como el que se implemento en este trabajo, puede ofrecer rendimiento comparables o iguales a los clusters dedicados implementados en disco.

6.5.5 BENCHMARK POVRAY

Este benchmark POVBENCH[1] consiste en trazar una escena en particular, con parámetros y opciones bien definidos, y tomar el tiempo

[1] POVBENCH, official PovRay Benchmarks, http://new.haveland.com/povbench/

de trazado de dicha escena. La escena consiste en una vasija, con una textura basada en una imagen de nubes, sobre un pedestal, frente a un escenario de paredes reflejantes. La escena es sencilla pero prueba todos los elementos básicos de un programa de trazado de rayos, como geometría solida constructiva, aplicación de texturas, y propiedades de materiales tales como reflexión y transparencia. Dicha escena se muestra a continuación.

Ilustración 22 Imagen obtenida con el benchmark POV-Ray

La ejecución de la escena se hizo en distribuido a través de OAR, se utilizo un script que divide la imagen a renderizar en franjas, genera una serie de trabajos y realiza el respectivo envió de dichos trabajos al servidor OAR, encargándose este de distribuir los trabajos a los nodos de computo. La misión de cada trabajo es renderizar cada una de las franjas y de enviarlas al servidor, aquí el script antes mencionado se encarga de unir todas las franjas recibidas, y así crear la imagen final. Esta prueba permite conocer el rendimiento de la infraestructura en la

ejecución de software distribuidamente, además de la interacción con el manejador de colas OAR, ya este es parte fundamental en esta prueba.

En esta prueba se utilizaron 70 equipos distribuidos en 3 salas de cómputo. En la grafica siguiente se muestra los tiempos consumidos en la renderización, para un determinado número de nodos. Las dimensiones de la imagen obtenida son 2048 x 1536.

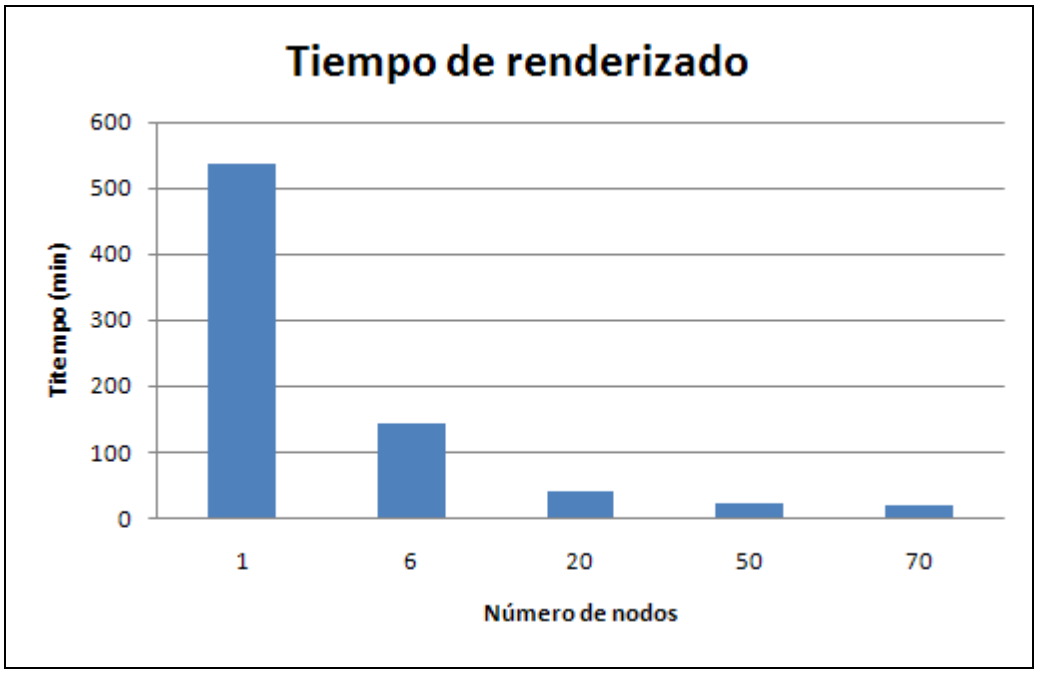

Ilustración 23 Tiempo de ejecución en el benchmark POV-Ray

Calculando el rendimiento teórico que se tendría utilizando 70 CPUs en condiciones ideales se obtiene:

Rendimiento Teórico = Tiempo total de la escena en una CPU/ # CPUs

= 536 min /70 = 7.65 min

Comparándose con el tiempo que se obtuvo para 70 maquinas que fue de 17 min, se observa que se aleja del rendimiento teórico por diferentes causas, dado que este tipo de aplicaciones se caracterizan

por la independencia casi total de los datos, hacen que factores como la velocidad de la red, no influyan significativamente en el rendimiento. En cambio factores que posiblemente pueden afectar el rendimiento, es la velocidad del manejador de colas para la asignación de los diferentes trabajos generados a los recursos disponibles. Otro factor es la granularidad del trabajo, dado que a veces no es posible generar una cantidad que sea múltiplo del número de recursos existentes, crean que llegado un determinado momento, algunos recursos se encuentren ociosos durante la prueba.

7. CONCLUSIONES

- ✓ Las pruebas demostraron que con suficiente memoria en los equipos de computo alrededor de 1 GB o 2GB, teniendo un sistema Linux implementado con servicios básicos y para aplicaciones que no demanden gran cantidad de datos, un cluster diskless no tiene diferencia alguna con un cluster con disco y dedicado en cuanto al rendimiento.

- ✓ Se pudo demostrar que es posible implementar una infraestructura de cálculo a través de la integración de los recursos no dedicados utilizando el periodo de tiempo en que estos están ociosos y que esta integración es transparente al usuario, tanto al usuario que presta su computadora, como al usuario que utiliza la infraestructura para suplir sus demandas de computo.

- ✓ El Esquema planteado permite generar un ambiente propicio para la realización de pruebas en sistemas distribuidos, reuniendo recursos no dedicados lo cual hace factible implementarlo en entornos donde es difícil conseguir máquinas dedicadas. Además de esto permite la integración de diferentes paquetes de software de propósito interdisciplinario, proporcionando ambientes configurables que pueden suplir diferentes necesidades

- ✓ La implementación de un sistema Linux que se almacenara completamente en RAM y la reducción de su tamaño, provoco que se redujera los tiempos de despliegue en un 49 % con respecto a Computemode y además se consiguiera un buen rendimiento en la ejecución de software en paralelo y distribuido.

- ✓ Este tipo de infraestructura implementada en el proyecto solo pretende explotar el tiempo ocioso de las maquinas durante un tiempo establecido como tal, por tanto no esta diseñado para recolectar ciclos ociosos de procesador en tiempos de inactividad del usuario. Aunque la herramienta utilizada provee un

mecanismo basado en el uso de maquinas virtuales y monitoreo de la actividad, esta no brinda ningún mecanismo que permita guardar el estado del trabajo lo cual es un muy importante si se quieren aprovechar los recursos en sus fragmentos de tiempo ocioso.

8. RECOMENDACIONES

- ✓ Creación o integración de un mecanismo que permita guardar el estado de un trabajo para su posterior reanudación, para evitar que ante una eventual caída, ya sea por causas eléctricas, el trabajo en proceso se pierda por completo, brindando una mejor calidad de servicio al usuario.
- ✓ Implementar una herramienta que permita la creación de sistemas Linux que puedan almacenarse en RAM, a partir de distribuciones preexistentes.

9. ANEXOS

9.1 INSTALACIÓN DE LOS COMPENENTES DEL SISTEMA

Para la instalación de los componentes de Computemode véase [15]

9.2 COMPILACIÓN DEL KERNEL

Para las instrucciones de compilación de un kernel Linux, se pude consultar [16]. El kernel que se implemento en el proyecto se compilo sin las siguientes características:

CARACTERISTICA	MODULOS ELIMINADOS
Procesor types and features	Paravirtualizacion
bus Options	Bus ISA, EISA, Generic PCI/EISA, MCA support, Natsemi, PCCard.
Network support:	Amateur radio suport, Irda infrared, Bluetooth subsystem support, Wireless , Wimax
Deviced drivers	Parallel port support, Plug and play support, Block devices execpto ram block, ATA , ATAPI support, SCSI support, RAID y LVM support, Macintosh device drivers, ISDN support, Telephony support, Inpurt device support, Souncard support, Infiniband support,
File systems	Ext2, Ext3, Reiserfs support, JFS filesystem support, XFS filesystem, GFS2 filesystem support , OCFS2 filesystem support, CD rom y DVD filesytem, DOS FAT NT filesystem.
Virtualization support	Todos

Tabla 3 Módulos eliminados en la compilación del kernel

Algunos de estas características por defecto son compiladas junto con el kernel y cargadas conjuntamente en RAM, otras son cargadas como módulos. La eliminación de estas características innecesarias produjo un kernel mas liviano y una menor cantidad de módulos, reduciendo el espacio ocupado por este en el sistema de archivos.

9.3 MANUAL DE UTLIZACIÓN DE LA INFRAESTRUCTURA

Esta parte solo hará descripción de el mecanismo de incorporación de aplicaciones y el ambiente grid, para una descripción de las otras herramientas como puede ser otras funcionalidades provistas por el manejador de colas OAR se puede consultar [17] y para mayor información de la interfaz de administración consultar [18].

El manejo de las aplicaciones en la infraestructura se realiza por medio de tres comandos *nodesapp*, *adminapp*, *appdeploy*. A continuación se describe el funcionamiento de cada uno de ellos.

Adminapp, es utilizado para mostrar las aplicaciones que hay en la base de datos, también permite creación y eliminación de aplicaciones.

Mostrar listado de aplicaciones

```
Computemode3:/home/camilo/Desktop/Perlcompute# ./Insert.pl
-l
Id: 5
Application name: MPICH
Configuration file: /opt/tools/
Id: 2
Application name: NAMD
Configuration file: /usr/bin/NamdInstall.sh
Id: 3
Application name: pov
Configuration file: /usr/bin/PovInstall.sh
Id: 1
Application name: default
Configuration file:
```

```
Crear una nueva aplicación
```

```
Computemode3:/home/camilo/Desktop/Perlcompute# ./Insert.pl
-name MPIBLAST -conf  /usr/bin/Mpiblast.sh
```

```
MPIBLAST
Application added succesfully
```

Remover una aplicación

Computemode3:/home/camilo/Desktop/Perlcompute# ./Insert.pl -r -id 6

The Application has been removed succesfully

nodesapp: permite la administración de las aplicaciones en los nodos al momento de la carga del sistema.

Mostrar un listado de las aplicaciones asignadas a los nodos

Computemode3:/home/camilo/Desktop/Perlcompute# ./nodesapp.pl -l

Hostname	Application
nodo11	default
nodo8	default
nodo9	default
nodo2	NAMD
nodo12	default
nodo1	Pov
nodo10	default
nodo7	default
nodo6	default
nodo3	default

Cambiar la aplicación asignada a un nodo

Computemode3:/home/camilo/Desktop/Perlcompute# ./nodesapp.pl -h nodo1 -id 2

Application updated succesfully

Desplegar una aplicación en nodos reservados

Se realiza una reservación de recursos tipo deploy

Computemode:/home/camilo# oarsub -I -t deploy -l /nodes=3

[ADMISSION RULE] Set default walltime to 7200.
[ADMISSION RULE] Modify resource description with type constraints
OAR_JOB_ID=11415
Interactive mode : waiting...
[2007-07-13 16:36:25] Starting...
Computemode:/home/camilo#

Se ejecuta el comando para el despliegue de las aplicaciones en los nodos antes reservados, para eso se utiliza la variable de entorno OARNODE_FILE.

Computemode:/home/camilo# ./AppDeploy –i 2 -f $OARNODE_FILE
Installing NAMD on nodo2 ……..
Installing NAMD on nodo3 ……..
Installing NAMD on nodo4 ……..
Installation process complete !!

Se conecta al trabajo

Computemode:/home/camilo# oarsub –C 11415

nodo2:/home/camilo#

9.4 TRABAJO DISTRIBUIDO

Reserva de recursos de forma distribuida

Computemode:/home/camilo# gridsub –s computemode, sistemas –l 2,4 -t “2009-10-22 11:00:00”

Making the reservation on the sites . . . [ok]

Your Job id is 6789 and 3476 use them to connect to the job

Conección al trabajo distribuido

Computemode:/home/camilo# gridconnect –id 6789, 3476

Setting the Grid environment . . . [ok]

Setting the Storage . . . [ok]

Conneting to the Job

nodo2:/home/camilo#

9.5 MODIFICACION SCRIPT INIT

El script *init* es ejecutado una vez el *ramdisk* es montado como sistema de archivos principal este script es comúnmente utilizado para la detección de hardware y carga de los respectivos módulos, además de establecer ciertos parámetros necesarios. El script fue modificado para permitir que el sistema de archivos se cargara totalmente en RAM, a continuación se presenta el código de dicho script.

```
#!/bin/sh

############################################################
####
# Se obtienen parametros de la linea del kernel
get_param()
{
      for I in $KERNELCMDLINE ; do
            eval `echo $I | grep "^$1="`
      done
}

echo "Loading , please wait"

[ -d /dev ] || mkdir -m 0755 /dev
[ -d /root ] || mkdir -m 0700 /root
[ -d /sys ] || mkdir /sys
[ -d /proc ] || mkdir /proc
[ -d /tmp ] || mkdir /tmp
mkdir -p /var/lock
mount -t sysfs -o nodev,noexec,nosuid none /sys
mount -t proc -o nodev,noexec,nosuid none /proc

# Note that this only becomes /dev on the real filesystem
if udev's scripts
# are used; which they will be, but it's worth pointing
out
tmpfs_size="10M"
```

```
if [ -e /etc/udev/udev.conf ]; then
      . /etc/udev/udev.conf
fi
mount -t tmpfs -o size=$tmpfs_size,mode=0755 udev /dev
[ -e /dev/console ] || mknod -m 0600 /dev/console c 5 1
[ -e /dev/null ] || mknod /dev/null c 1 3
> /dev/.initramfs-tools
mkdir /dev/.initramfs

# Set modprobe env
export MODPROBE_OPTIONS="-qb"

  # Export relevant variables
  export ROOT=
  export ROOTDELAY=
  export ROOTFSTYPE=
  export break=
  export init=/sbin/init
  export quiet=n
  export readonly=y
  export rootmnt=/root.new
  export debug=
  export panic=
  export blacklist=
  export resume_offset=
  export FUENTE="/root.new/"

  . /scripts/functions

  depmod -a

  maybe_break top

  # Don't do log messages here to avoid confusing usplash
  run_scripts /scripts/init-top

  maybe_break modules
  log_begin_msg "Loading essential drivers..."
  load_modules
  log_end_msg
```

```
  maybe_break premount
  [ "$quiet" != "y" ] && log_begin_msg "Running
  /scripts/init-premount"
  run_scripts /scripts/init-premount
  [ "$quiet" != "y" ] && log_end_msg

  maybe_break mount

  echo "testing modprobe nfs"
  modprobe nfs

  echo "testing modprobe af_packet"

  modprobe af_packet

  echo "Loading varible Cmdline of kernel "
  KERNELCMDLINE=`cat /proc/cmdline `
  export KERNELCMDLINE

  get_param HOSTNAME

  get_param IP

# configuracion de la interfaz de red

ifconfig eth0 $IP netmask 255.255.255.0  up

mkdir /root.new/
echo "Mounting file system "

get_param NFSROOT
nfsmount -o ro,nolock $NFSROOT:/fsdiskless /root.new/
echo "Mounting woking directory "
nfsmount $NFSROOT:/home/ /home/

echo "Setting up root file system"
/root.new/tools/Buildroot.sh
umount root.new/

# Transferencia de la aplicacion
echo "Setting up application"
```

/usr/bin/diskless_application.sh

```
# Setting the hostname
echo "Setting up the hostname"
echo $HOSTNAME >/etc/hostname

echo "Unmounting file systems : root.new, proc, sys "
umount root.new/
umount proc/
umount sys/

#Ejecucion del init, el primer proceso en ejecucion en
todo sistema Linux
exec /sbin/init <dev/console >dev/console
```

10. BIBLIOGRAFIA

[1] Gillen A, Kusnetzky D, Sayana A, Dayal H, Hingley M. Windows Operating Environment Forecast and Analysis. IDC publisher, document 24827, 2001.

[2] Bruno Richard, Philippe Augerat. I-Cluster: Intense computing with untapped resources. HP Laboratories Grenoble April 4 2002.

[3] Pxe specification, The Preboot Execution Environment specification v2.1 published by Intel y Systemsoft. September 20 1999.

[4] James H, Laros III, Lee H ward. Implementing Scalable Disk-less Clusters using the Network File System (NFS). Sandia National labs. October 30 2003.

[5] Baris guler, Munira Hussain, Tau Leng, Victor Mashayekhi. The advantages of Diskless HPC Clusters Using NAS. Dell power solutions November 2002.

[6] Kernel org Home page. What is ramfs?. Disponible en : http://www.kernel.org/doc/Documentation/filesystems/ramfs-rootfs-initramfs.txt. Consultada el (26 julio 2009).

[7] Landley, Rob (15 March 2005), Introducing initramfs, a new model for initial RAM disk, http://www.linuxfordevices.com/c/a/Linux-For-Devices-Articles/Introducing-initramfs-a-new-model-for-initial-RAM-disks/.

[8] TobiasWittwer, An Introduction to Parallel Programming, VSSD 2006.

[9]Turn your desktop computer into a high-peroformance cluster computing cluster, Cluster to Cluster, MAYANK SHARMA , Linux Magazine junio 2009.

[10] Mark Baker, Rajkumar Buyya, Domenico Laforenza. Grids and Grid Technologies for wide-area distributed computing. University of Portsmouth, University of Melbourne, Instituto del Consiglio Nazionale delle Ricerche. Septiembre 2002.

[11]The Linux Information Headquaters, Memory management, http://www.linuxhq.com/guides/TLK/mm/memory.html, consultada el 12 septiembre 2009.

[12] *J. Zuluaga , A. Ospina . Installation and Configuration of a Cluster-room as a Low Cost Solution for the Access to Distributed Computing Technologies in Latin America*. Grupo de Fisica y Astrofisica Computacional, Universidad de Antioquia, Colombia; Grupo de Microelectronica, Universidad Pontifica Bolivariana, Colombia.

[13] Eliza Guardo Martínez, Infraestructura Grid basada en virtualización Departamento de Ingeniería de sistemas y computación Universidad de los Andes, 2008.

[14] Iptables tutorial 1.2.2, http://iptables.rlworkman.net/iptables-tutorial.html, Consultada el 16 de septiembre 2009.

[15] Computemode on top of debian etch, http://computemode.imag.fr/mediawiki/index.php/ComputeMode_on_top_of_Debian_Etch, Consultada el 15 de junio 2009.

[16] Compiling the Linux kernel, http://linuxgazette.net/111/krishnakumar.html, Consultada el 17 de Julio 2009.

[17] Manual OAR, http://oar.imag.fr/admins/admin_documentation.html, Consultada el 20 de Agosto 2009.

[18] Manual Computemode, http://computemode.imag.fr/old/wrapper.php?key=manual15, Consultada el (21 de Octubre de 2009).

[19]Sistema de Arhivos Temporal, http://lxr.linux.no/linux/Documentation/filesystems/tmpfs.txt, consultada el 11 agosto 2009.

[20]A. Petiet, R. C. Whaley, J . Dongarra, A. Cleary, HPL – A Portable Implementation of High-Performance Linpack Benchmark for Distributed- Memory Computers, 2001, http://www.netlib.org/benchmark/hpl/

[21] Resource Management System for High Peformance Computing, http://oar.imag.fr/index.html, consultada el 10 de septiembre 2009.