Universidad Carlos III de Madrid
Escuela Politécnica Superior

Ingeniería en Informática
Proyecto Fin de Carrera

Diseño y despliegue de un cluster Linux para el soporte a la docencia

Autor: Jaime Pons Bailly-Bailliere
Tutores: Francisco Javier García Blas
Óscar Pérez Alonso

Mayo, 2009

Agradecimientos:

En primer lugar tengo que agradecer muy especialmente a mis padres el haberme dado todos los medios y la motivación que a veces me faltaba para permitirme llegar a donde he llegado. Gracias de todo corazón por ser como sois.

En segundo lugar quiero agradecer a Cris, mi pareja, que desde que se hizo un hueco en mi vida, la haya cambiado para siempre. Gracias por estar tan pendiente de mí en los buenos y en los malos momentos, por estar ahí siempre que lo he necesitado.

También tengo que agradecer enormemente a todo el Laboratorio del Departamento de Informática, por facilitarme los medios para el despliegue de este proyecto, y a todos, becarios y técnicos, que me han echado una mano cuando lo he necesitado. Ahí os dejo el muerto… ¡Espero que lo disfrutéis mucho!

Dentro del Laboratorio tengo que dar las gracias a Óscar, ya que sin una persona como él dentro del Laboratorio este proyecto no se hubiese terminado nunca. Sinceramente me ha encantado trabajar codo con codo con una persona como tú y espero haber aprendido la mitad de las cosas que sabes.

Gracias a todas las personas que han pasado por mi "vida universitaria", ya que para bien o para mal todos vais a formar parte de mis recuerdos.

Gracias a todas aquellas personas con las que he trabajado, haciendo prácticas y memorias interminables. Especialmente Antonio, una de las pocas personas que me llevo de la Universidad, considerándole amigo de verdad.

Por último, y no por ello menos importante, a mi tutor Javi, ya que he visto cómo te has implicado en mi proyecto, aún siendo en una de las peores épocas universitarias. Gracias por aguantarme estas últimas semanas de continuos correos y conversaciones agobiantes por el Messenger.

Gracias sinceramente a todos mis amigos y a todas las personas que de verdad se alegran por mí y por este día.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 INTRODUCCIÓN

En este capítulo se describen de forma general los orígenes del proyecto, dónde se explican las razones por las que se ha considerado realizar el proyecto y los objetivos que se desean cumplir mediante el desarrollo del mismo. Además se incluye una sección en la que se define la estructura que va a seguir el documento, así como los acrónimos usados durante todo el documento.

1.1 Motivación

A lo largo del curso académico se desarrollan en la Universidad estudios e investigaciones por parte del cuerpo docente, así como trabajos relacionados con las asignaturas por parte de los alumnos. Para dichas tareas es preciso hacer uso de una gran cantidad de recursos, tanto tecnológicos como logísticos. Es por este motivo que surge la necesidad de satisfacer estas necesidades mediante la implantación de un sistema cluster.

La utilización de los equipos disponibles en las aulas docentes tiene sus ventajas y sus inconvenientes, como se va a explicar a continuación:

Ventajas:

- Los resultados de las ejecuciones son obtenidos en el **momento** en que terminan.
- En caso de utilizar una aplicación **visual**, se obtiene la vista del programa en tiempo real.
- Las **modificaciones** oportunas pueden ser realizadas en el momento.

Inconvenientes:

- Los recursos de la Universidad son **limitados** y hay multitud de usuarios que requieren de equipos para realizar sus trabajos, por lo que no siempre se puede disponer de un equipo o varios cuando se necesita.
- Cuando se desea ejecutar aplicaciones que hacen uso de uno o más equipos, no se puede garantizar la disponibilidad de estos.
- Las aplicaciones ejecutadas por los distintos usuarios pueden precisar desde un gran número de recursos (número de equipos, tiempo de ejecución, una arquitectura determinada, etc.) hasta distintas configuraciones desplegadas (sistema operativo, software pre-instalado, etc.).

Al no existir en estos momentos un sistema de estas características, y al considerarlo altamente beneficioso tanto para el claustro docente como para los alumnos de la Universidad, se convierte en un recurso necesario para el centro.

Se pretende ofrecer un sistema abierto y amigable para los alumnos y el personal docente del Centro y que les permita tener acceso constante a los datos de su cuenta y a un entorno de trabajo en el que lanzar sus programas, prácticas o investigaciones

1.2 Objetivos

El objetivo principal de este proyecto es el de implantar un sistema de ejecución de trabajos en paralelo para que pueda ser utilizado por todo personal perteneciente a la Universidad Carlos III de Madrid y que además posea una cuenta en las aulas del Laboratorio del Departamento de Informática.

Por tanto los servicios que deben estar en funcionamiento para el correcto funcionamiento del sistema son:

- Almacenamiento de datos compartidos.
- Servicio de cuentas compartidas en red.
- Acceso mediante *ssh* al servidor para servicios de administración y mantenimiento.
- Servicio de resolución de direcciones.
- Administración de ordenadores en tipología de cluster.
- Administración y lanzamiento de trabajos en cluster.

1.3 Estructura del documento

En este apartado se expone la estructura que va a regir el desarrollo de este documento, la cual consta de siete capítulos y un apéndice.

En el primer capítulo, con título "*Introducción*" se explican los diversos motivos que han llevado a considerar la realización de este proyecto, así como los objetivos que se pretenden alcanzar durante el desarrollo del mismo.

En el segundo capítulo, con título "*Estado de la cuestión*" se realiza en primer lugar una visión general de todas las ideas y aplicaciones que se desean estudiar y utilizar, y mas tarde se explica de forma más exhaustiva el motivo por el cual se cree conveniente utilizar dicha tecnología para llevar a cabo el desarrollo de este proyecto.

El tercer capítulo, con título "*Análisis*", se hace un estudio detallado de los elementos necesarios para cumplir con los objetivos marcados en el primer apartado del documento. Además se realiza un análisis de requisitos que van a ser referencia obligada para llevar a cabo cada punto del desarrollo del proyecto.

El cuarto capítulo, "*Diseño del cluster*" hace una exposición general de la estructura que va a seguir la arquitectura cluster del proyecto, haciendo énfasis en los aspectos más importantes que deben ser desarrollados.

El quinto capítulo, denominado "*Despliegue del cluster*" trata de explicar de forma comprensible los pasos que se han seguido para poder instalar y configurar cada una de las aplicaciones necesarias para conseguir el correcto funcionamiento del cluster.

En el sexto capítulo, con título "*Evaluación*" se exponen los resultados que se han obtenido al ejecutar diversas pruebas en el cluster y se realizan comparaciones con los resultados que se obtienen al ejecutar las mismas pruebas en los superordenadores más potentes del mundo.

En el séptimo y último capítulo, con título "*Conclusiones y trabajos futuros*" se exponen las conclusiones finales a las que se ha llegado después de la realización del proyecto, haciendo una revisión de los objetivos iniciales y estudiando si se han cumplido debidamente cada uno de ellos. Además se indican una serie de futuros trabajos que se pueden realizar como ampliación al desarrollo de este proyecto. Por último se exponen los recursos necesarios, tanto tecnológicos como de personales, para el desarrollo del mismo.

En el primer y único apéndice llamado "*Apéndice I*" se explica cómo se han estructurado los diferentes scripts de administración que han sido implementados para la instalación y el mantenimiento de diversas aplicaciones, además de los archivos de configuración más importantes.

En "*Referencias*" se indican las páginas y los documentos a las que se ha hecho algún tipo de referencia a lo largo del presente documento.

Finalmente en el capítulo de "*Acrónimos y abreviaturas*" se pueden encontrar las definiciones de los acrónimos más importantes que aparecen a lo largo de este documento.

1.4 Características del documento

En este punto es necesario entablar las particularidades de este documento. Este documento tiene las siguientes características:

- Toda mención a código fuente o sentencias estarán en letra Courier.
- Los títulos vendrán en negrita y en estilo de letra Times New Roman.
- El resto de texto vendrá dado en letra Times New Roman con un tamaño de 12 puntos y un interlineado de un espacio y medio.
- Las palabras y puntos más destacados de cada página estarán resaltadas en cursiva y negrita.

1.5 Acrónimos y abreviaturas

A continuación se especifican los principales acrónimos que se pueden encontrar a lo largo del presente documento.

Acrónimo	Definición
TCP	Transmission Control Protocol
IP	Internet Protocol
BSD	Berkeley Software Distribution
NIS	Network Information Service
DNS	Domain Name Server
LAN	Local Area Network
MPI	Message Passing Interface
PBS	Portable Batch System
iSCSI	Internet Small Computers System Interface
PVFS	Parallel Virtual File System
NFS	Network File System
OCFS	Oracle Cluster File System
Intel-VT	Intel Virtualization Technologies
AMD-V	AMD Virtualization
BLAS	Basic Linear Algebra Subroutines
Mbps	Megabits por segundo
Gbps	Gigabits por segundo
TFT	Thin Film Transistor
Ghz	Gigaherzios

2 ESTADO DE LA CUESTIÓN

Para desplegar correctamente un sistema de computación cluster, se debe en primer lugar realizar un estudio completo del dominio y las aplicaciones que se pueden utilizar y su **funcionalidad**, con el objetivo de que dicho despliegue responda a las distintas necesidades de las personas a las que va destinado.

En este capítulo se exponen de forma detallada las diferentes tecnologías software disponible actualmente, y que pueden ser utilizadas para el desarrollo de este proyecto. En primer lugar se realiza una exposición general de la **tecnología cluster**, comenzando por su historia, desde sus inicios, sus precursores y su desarrollo.

Por último se van a estudiar los diferentes **sistemas de ficheros** que se han pensado utilizar para el desarrollo de este proyecto. Uno de los objetivos que persigue este proyecto es realizar una comparativa experimental del comportamiento de varios sistemas de ficheros, y por lo tanto decidir finalmente cual de ellos es el más adecuado para ser desplegado en una arquitectura de tipo cluster.

Este estudio previo ha servido para escoger finalmente la tecnología que más se adecua a las necesidades del proyecto.

2.1 Cluster

El origen del término y uso de este tipo de tecnología es desconocido pero se puede considerar que comenzó entre finales de los años 50 y principios de los años 60.

La base formal del término, en cuanto a Ingeniería Informática se refiere, está relacionado con trabajos en paralelo de cualquier tipo. En 1967, Gene Amdahl publicó lo que ha llegado a ser considerado como el inicio del procesamiento paralelo: la **Ley de Amdahl** [1]. Esta ley que describe matemáticamente el aceleramiento que se puede esperar paralelizando cualquier serie de tareas al ser aplicadas sobre una arquitectura paralela.

La historia de los primeros grupos de computadoras está fuertemente ligada a la historia de los inicios de las redes. Una de las principales motivaciones para el desarrollo de una red era **enlazar los distintos recursos** que ofrece un conjunto de computadoras.

Utilizando el concepto de red de conmutación de paquetes, el proyecto ARPANET [2] logró crear en 1969 lo que fue posiblemente la primera red de ordenadores básica basada en un cluster de ordenadores formado por cuatro tipos de centros informáticos. El proyecto ARPANET creció y se convirtió en lo que es ahora Internet, que se puede considerar como "la madre de todos los clusters" (como la unión de casi todos los recursos de cómputo, incluidos los clusters, que pasarían a ser conectados).

También estableció el paradigma de uso de computadoras cluster en el mundo de hoy que consiste en el uso de las redes de conmutación de paquetes para realizar las comunicaciones entre procesadores localizados en los marcos de otro modo desconectado.

La construcción de PC's por los clientes y grupos de investigación se desarrolló a la vez que la organización de las redes y el sistema operativo Unix desde principios de la década de los años 70. Estos desarrollos dieron sus frutos, sacando a la luz proyectos

como TCP/IP [3] y el proyecto de la Xerox PARC formado por protocolos basados en redes de comunicaciones. El núcleo del sistema operativo fue construido en 1971.

Sin embargo, no fue hasta alrededor de 1983, cuando los protocolos y herramientas para el trabajo remoto, comenzaron a facilitar la distribución y uso compartido de archivos y recursos (en gran medida dentro del contexto de BSD Unix [4], e implementados por Sun Microsystems).

El primer producto comercial de tipo cluster fue ARCnet [5], desarrollado en 1977 por Datapoint. No obtuvo un éxito comercial y los cluster no consiguieron tener éxito hasta que en 1984 VAXcluster [6] fue desarrollado con el sistema operativo VAX/VMS. El ARCnet y VAXcluster no sólo son productos que apoyan la computación paralela, sino que además también comparten los sistemas de archivos y dispositivos periféricos. La idea era proporcionar las ventajas del procesamiento paralelo, al tiempo que se mantiene la fiabilidad de los datos y el carácter singular.

Otros dos principales clusters comerciales desarrollados más adelante fueron el Tandem Himalaya [7] (alrededor de 1994 con productos de alta disponibilidad) y el IBM S/390 Parallel Sysplex [8] (también alrededor de 1994, principalmente destinado a empresas).

La historia de los clusters de ordenadores no estaría completa sin señalar el papel fundamental desempeñado por el desarrollo del software de **Parallel Virtual Machine** (PVM) [9]. Este software de código abierto y basado en comunicaciones TCP/IP permitió la creación de un superordenador virtual, un cluster de computación de alto rendimiento (High Performance Computing). De forma libre los clusters heterogéneos han constituido la cima de este modelo logrando aumentar rápidamente en Flops y superando con creces la disponibilidad incluso de los más caros superordenadores.

PVM y el el empleo de PCs de bajo coste llevó, en 1993, a un proyecto de la NASA para construir supercomputadoras de clusters.

En 1995, la invención de la arquitectura de tipo "*beowulf*" [10], una granja de computación diseñado en base a un producto básico de la red con el objetivo específico de "ser un superordenador" capaz de realizar de forma eficiente cálculos paralelos. Esta invención estimuló el desarrollo independiente de la computación Grid como una

entidad, a pesar de que el estilo Grid giraba en torno al del sistema operativo Unix y el Arpanet.

Beneficios de la Tecnología Cluster

Las aplicaciones paralelas escalables requieren: buen rendimiento, comunicaciones que dispongan de gran ancho de banda y baja latencia, redes escalables y acceso rápido a los archivos. Un cluster puede satisfacer estos requisitos usando los recursos que tiene asociados a él.

Los cluster ofrecen las siguientes **características** a un costo relativamente bajo:

- Alto Rendimiento.
- Alta Disponibilidad.
- Alta Eficiencia.
- Escalabilidad.

La tecnología cluster permite a las organizaciones incrementar su capacidad de procesamiento usando tecnología estándar, tanto en componentes de hardware como de software, que pueden adquirirse a un costo relativamente bajo.

Clasificación de los Clusters

El término cluster tiene diferentes connotaciones para diferentes grupos de personas. Los tipos de clusters, establecidos en base al uso que se de a los clusters y los servicios que ofrecen, determinan el significado del término para el grupo que lo utiliza. Los clusters pueden **clasificarse** en base a sus características. Los tres tipos de cluster que se pueden tener son:

- **Alto rendimiento**: Son clusters en los cuales se ejecutan tareas que requieren de gran capacidad computacional, grandes cantidades de memoria, o ambos a la vez. El llevar a cabo estas tareas puede comprometer los recursos del cluster por largos periodos de tiempo.
- **Alta disponibilidad**: Son clusters cuyo objetivo de diseño es el de proveer disponibilidad y confiabilidad. Estos clusters tratan de brindar la máxima

disponibilidad de los servicios que ofrecen. La confiabilidad se provee mediante software que detecta fallos y permite recuperarse frente a los mismos, mientras que en hardware se evita tener un único punto de fallos.

- **Alta eficiencia**: Son clusters cuyo objetivo de diseño es el ejecutar la mayor cantidad de tareas en el menor tiempo posible. Existe independencia de datos entre las tareas individuales. El retardo entre los nodos del cluster no es considerado un gran problema.

Los clusters pueden ser clasificados también como Clusters de IT **Comerciales** (Alta disponibilidad, Alta eficiencia) y Clusters **Científicos** (Alto rendimiento). A pesar de las discrepancias a nivel de requerimientos de las aplicaciones, muchas de las características de las arquitecturas de hardware y software que están por debajo de las aplicaciones en todos estos clusters, son las mismas. Más aún, un cluster de determinado tipo, puede también presentar características de los otros.

2.1.1 Cluster de tipo Beowulf

El cluster encuadrado en la tipología *Beowulf* posee una arquitectura basada en **multicomputadoras** y suele ser utilizado para computación paralela. Este sistema consta de un nodo maestro y uno o más nodos esclavos conectados a través de una red ethernet u otra topología de red. Esta construido con componentes de hardware comunes en el mercado, similar a cualquier PC, y adaptadores de ethernet y switches estándares. Como no contiene elementos especiales es fácilmente reproducible, y por tanto es sencillo implementarlo con unos mínimos recursos básicos.

Una de las diferencias principales entre un cluster *Beowulf* y un cluster de estaciones de trabajo (COW, *cluster of workstations*) [11] se basa en que la visión que ofrece un cluster *Beowulf* se similar a una sola máquina que muchas estaciones de trabajo. En la mayoría de los casos, los nodos de cómputo no tienen dispositivos de E/S como monitores o teclados y son accedidos solamente vía remota o por terminal serial.

El nodo maestro controla el cluster y presta servicios de sistemas de archivos a los nodos esclavos. Es también la consola del cluster y la conexión hacia el mundo exterior. Los sistemas grandes *Beowulf* pueden tener más de un nodo maestro, y otros nodos dedicados a diversas tareas específicas, como por ejemplo, consolas o estaciones de

supervisión. En la mayoría de los casos los nodos esclavos de un sistema *Beowulf* son estaciones simples. Los nodos son configurados y controlados por el nodo maestro, y hacen solamente lo que éste le indique.

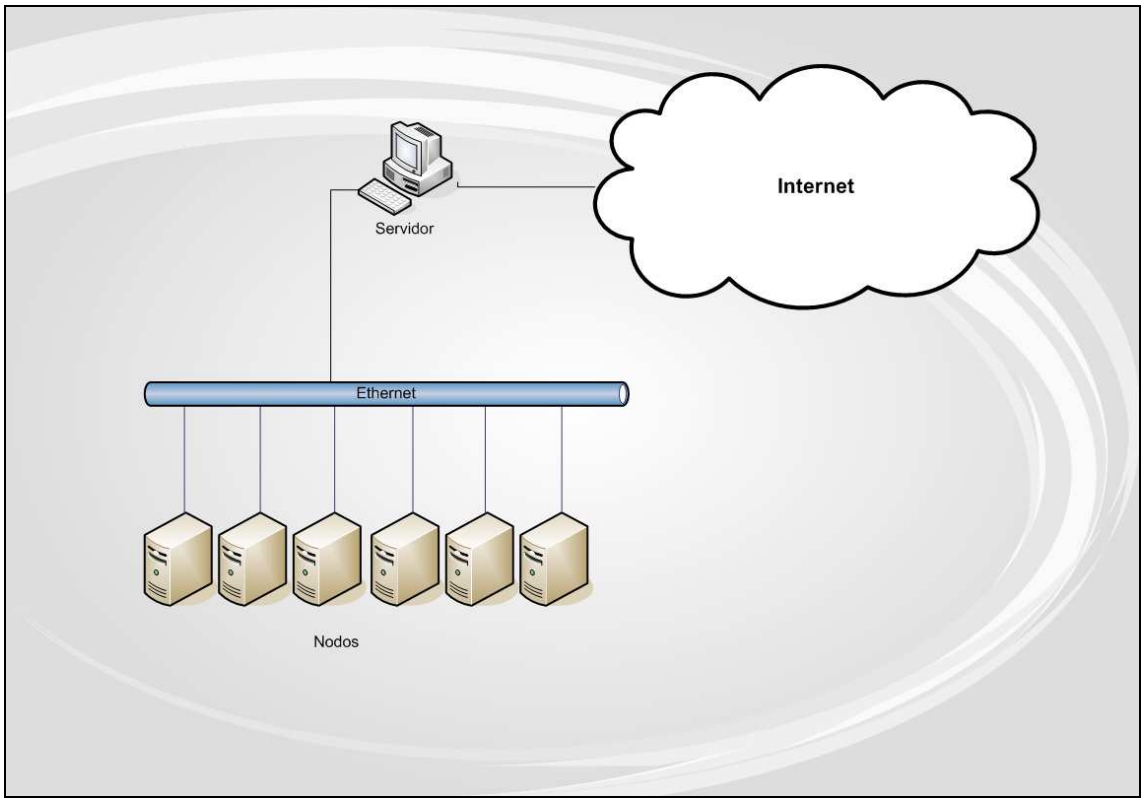

Figura 1: Esquema general de un cluster

2.1.2 Linux Cluster Manager (LCM)

La herramienta LCM [12] proporciona una interfaz gráfica desde la que manejar de forma fácil y sencilla múltiples equipos instalados con sistemas Linux desde un solo terminal y de forma centralizada. Está diseñado principalmente para clusters que siguen el formato de *Beowulf*, y proporciona muchos y muy útiles rasgos para la administración de un sistema general. Incluso ofrece posibilidades no específicas de Linux, como la **instalación de imágenes** a nivel de bloque.

Gracias a la herramienta LCM, las imágenes (copias del sistema operativo) se implantan en los nodos de forma rápida y sencilla, debido a que la propia herramienta se encarga de modificar la IP, el nombre de la máquina (*hostname*) e independiza las aplicaciones de todos los nodos automáticamente. Usando una imagen basada en ficheros se puede cambiar el dispositivo deseado, la capacidad del sistema de ficheros o incluso el tipo de sistema de ficheros. Por tanto la principal propiedad de esta

herramienta es que permite reinstalar la imagen en un nodo. Una vez instalada, la computadora estará lista para utilizarse.

La creación de imágenes se realiza desde una máquina en funcionamiento, a partir de la cual el LCM almacena con las características propias de una imagen genérica.

Todas las operaciones, hasta las más complicadas, se realizan en modo visual, señalando y pulsando las opciones deseadas. Gracias a estas características se consigue no tener que recordar extraños comandos o configuraciones complicadas.

Características generales:

- Fácil uso de la interfaz gráfica para todo tipo de operaciones.
- Información del estado de todos los nodos en tiempo real.
- Conexión a cada nodo individual según el protocolo escogido por el usuario (*ssh*[13], *rsh*[14], *rlogin*[15], etc).
- Informes de los procesos que se están ejecutando en el cluster.

Características de las imágenes:

- Las imágenes de los nodos recién instalados tienen la información de la IP y el nombre del equipo (*hostname*) cambiado automáticamente al arrancar por primera vez.
- Instalación de imágenes a nivel de bloque, que puede ser usado por cualquier sistema operativo basado en x86.
- Instalación de imágenes a nivel de fichero.
- Las imágenes pueden ser modificadas, como el dispositivo en el que se instalará, el tamaño o el tipo del sistema de ficheros.

Características de monitorización:

- Monitorización en tiempo real del estado de los nodos, tanto de CPU como de red.
- Gráficas escalables.
- Posibilidad de monitorizar todos los nodos o un conjunto de elllos.

Características de los scripts:

- Ejecutar un script en todo el cluster o solo en un conjunto de ellos.
- Conectar con un protocolo seleccionado por el usuario, con autenticación de seguridad.
- Acceso mediante línea de comandos o por interfaz gráfica.
- Pueden ser incorporados a scripts externos.

2.1.3 NIS

Network Information Service [16] (conocido por su acrónimo NIS, cuya traducción prodría ser Sistema de Información de Red), es el nombre de un protocolo de servicios de directorios cliente/servidor desarrollado por Sun Microsystems. Su objetivo es el **envío de información** de configuración en sistemas distribuidos, como podrían ser el nombre de usuarios y equipos.

Originalmente NIS era conocido como Páginas Amarillas (Yellow Pages), o YP, aunque todavía se utiliza para referirse a él. Desafortunadamente, ese nombre es una marca registrada de British Telecom, que exigió a Sun abandonar esa denominación. Sin embargo YP permanece como prefijo en los nombres de la mayoría de las órdenes relacionadas con NIS, como “ypserv” e “ypbind”.

DNS (Domain Name Server) [17] sirve un rango limitado de información, siendo la más importante la correspondencia entre el nombre de nodo y la dirección IP. Para otros tipos de información, no existe un servicio especializado así. Por otra parte, si sólo se administra una pequeña LAN (Local Area Network) [18] sin conectividad a Internet, no parece que merezca la pena configurar DNS. Ésta es la razón por la que Sun desarrolló el Sistema de Información de Red (NIS). NIS proporciona prestaciones de

acceso a bases de datos genéricas que pueden utilizarse para distribuir, por ejemplo, la información contenida en los ficheros "passwd" y "groups" a todos los nodos de su red. Realizar estas acciones provoca que la red parezca un sistema individual, con las mismas cuentas en todos los nodos. De manera similar, se puede usar NIS para distribuir la información de nombres de nodo contenida en "/etc/hosts" a todas las máquinas de la red.

Implementaciones

Hoy NIS está disponible prácticamente en todas las distribuciones de Unix, e incluso existen implementaciones libres. BSD Net-2 publicó una que ha sido derivada de una implementación de referencia de dominio público donada por Sun. El código de la biblioteca de la parte cliente de esta versión existe en la *libc* de Linux desde hace mucho tiempo, y los programas de administración fueron portados hace tiempo a Linux.

2.1.4 Gestores de colas de trabajo

En este punto se han estudiado dos gestores de trabajo para clusters, uno ya utilizado por el Departamento de Informática, como *Torque* [19], y otro más completo pero a su vez más complejo como es *Condor* [20].

2.1.4.1 Torque

Torque es un sistema administrador de tareas de código abierto que proporciona un control total sobre lanzamientos de trabajos por lotes y en nodos de computación distribuida. Ha sido desarrollado gracias al esfuerzo de la comunidad basada en el original proyecto PBS (Portable Batch System) [21] y con más de 1200 parches. Ha incorporado importantes avances en las áreas de escalabilidad, tolerancia a fallos, y ha ampliado las características ofrecidas por distintas instituciones como NCSA (National Center of Supercomputing Applications), Ohio Supercomputer Center (OSC), Pacific Northwest National Laboratory (PNNL) o TeraGrid, y muchas otras organizaciones de vanguardia en computación de alto rendimiento. Esta versión puede ser modificada y redistribuida libremente, pero siempre sujeta a las limitaciones de la licencia incluidas.

Torque utiliza para su ejecución *OpenPBS* [22], la versión original del sistema portable por lotes PBS y se trata de un sistema de colas desarrollado por la NASA a principios de 1990. *OpenPBS* funciona en redes multiplataforma en entornos UNIX.

Características generales:

Torque proporciona mejoras en los estándares *OpenPBS* en las siguientes áreas:

- Tolerancia a fallos adicional sobre condiciones de fallo controladas y manejo de las condiciones de "salud" de los nodos mediante scripts.
- Interfaz de **planificación:**
    - Extensión de las prestaciones de la interfaz de búsqueda con información adicional y más precisa.
    - Extensión del control de la interfaz que permite al programador un mayor control sobre el comportamiento y los modos de empleo.
    - Permite la recopilación de estadísticas de trabajos completados.
- **Escalabilidad**:
    - Mejorado significativamente el modelo de comunicación.
    - Capacidad para manejar grandes grupos (más de 15 procesadores)
    - Capacidad para manejar trabajos más grandes (más de 2000 procesadores)
    - Capacidad para soportar mensajes del servidor más grandes.
- **Usabilidad**:
    - Se amplía la variedad de archivos de *log*.
    - Logs más fáciles de leer.

2.1.4.2 Condor

El objetivo del proyecto *Condor* es desarrollar, implementar, implantar, y evaluar mecanismos y políticas de soporte para obtener una computación de alto rendimiento (High throughput Computing) en grandes colecciones de recursos computacionales distribuidos. Orientado a competir tanto tecnológica como sociológicamente con varios entornos de computación, el equipo de *Condor* ha estado construyendo herramientas software que permitan a científicos e ingenieros incrementar su rendimiento computacional.

Condor es un sistema especializado en organizar la carga de trabajo en sistemas de computación intensivos. Como otros sistemas completos por lotes, *Condor* proporciona un mecanismo de colas de trabajo, planificando políticas, esquemas de prioridades, monitorización de recursos, y administración de recursos. Los usuarios envían sus trabajos, ya sean paralelizables o no, a *Condor* y el propio programa los emplaza en la cola de ejecución, eligiendo cuando y donde poner a ejecutar los trabajos basándose en la política asignada, monitorizando el proceso cuidadosamente y finalmente informando al usuario cuando finaliza la ejecución.

Mientras que la funcionalidad que proporciona es similar a los tradicionales sistemas de colas por lotes, la arquitectura de *Condor* ofrece como novedad la posibilidad de ejecutar con éxito trabajos donde los sistemas de planificación tradicionales fallan. Condor puede ser usado para manejar un cluster con nodos de computación dedicada (como el cluster de tipo "*Beowulf*"). Además los mecanismos únicos de *Condor* le permiten aprovechar eficazmente los recursos de CPU no utilizados por equipos de mesa. Por ejemplo, *Condor* puede ser configurado para usar tan solo las máquinas que en un momento dado no se estén utilizando ni el teclado ni el ratón. El sistema debería detectar que una máquina deja de estar disponible (cuando se le pulsa una tecla, por ejemplo) y en muchas circunstancias es capaz de producir una parada en la ejecución y migrar el trabajo a una máquina que en ese momento esté ociosa.

Condor no requiere un sistema de ficheros compartido entre las máquinas ya que si no están disponibles los datos en modo compartido, *Condor* puede transferir los archivos de trabajo en nombre del usuario o transparentemente redirigir todo el trabajo mediante peticiones de E/S a la máquina disponible. Por tanto, *Condor* puede usarse

perfectamente para combinar el poder computacional de todos los recursos de una organización.

2.1.5 Interfaz de paso de mensajes (MPI)

La Interfaz de Paso de Mensajes [23] (conocido ampliamente como MPI, siglas en inglés de Message Passing Interface) es un paradigma de comunicación entre computadoras. MPI define el estándar para la comunicación entre los nodos que ejecutan un programa en un sistema de memoria distribuida. Las implementaciones en MPI consisten en un conjunto de bibliotecas de rutinas que pueden ser utilizadas en programas escritos en los lenguajes de programación "C", "C++", "Fortran" y "Ada". La ventaja de MPI sobre otras bibliotecas de paso de mensajes, consiste en que los programas que utilizan la biblioteca son portables (dado que MPI ha sido implementado para casi toda arquitectura de memoria distribuida), y rápidos, (porque cada implementación de la librería ha sido optimizada para el hardware en la cual se ejecuta).

El paso de mensajes es una técnica empleada en programación concurrente para aportar sincronización entre procesos y permitir la exclusión mutua, de manera similar a como se hace con los semáforos, monitores, etc.

Su principal característica es que no precisa de memoria compartida (aunque alguna de las implementaciones hagan uso de ella), por lo que es muy importante en la programación para sistemas distribuidos. Los elementos principales que intervienen en el paso de mensajes son el proceso que envía, el que recibe y el mensaje.

Dependiendo de si el proceso que envía el mensaje espera a que el mensaje sea recibido, se puede hablar de paso de mensajes síncrono o asíncrono. En el paso de mensajes asíncrono, el proceso que envía, no espera a que el mensaje sea recibido, y continúa su ejecución, siendo posible que vuelva a generar un nuevo mensaje y a enviarlo antes de que se haya recibido el anterior. Generalmente empleando este sistema, el proceso que envía mensajes solo se bloquea o para, cuando finaliza su ejecución, o si el buzón está lleno. En el paso de mensajes síncrono, el proceso que envía el mensaje espera a que un proceso lo reciba para continuar su ejecución. Dentro del paso de mensajes síncrono se engloba a la llamada a procedimiento remoto, muy popular en las arquitecturas cliente/servidor.

2.1.5.1 MPICH

MPICH [24] es una de la implementaciones más conocidas de MPI. MPICH se distribuye de forma gratuita y está disponible para la mayoría de Unix (incluyendo Linux y Mac OS X) y Microsoft Windows. MPICH es una implementación de MPI optimizada para entornos homogéneos lo que proporciona un mayor rendimiento en el paso de mensajes entre nodos.

La implementación original de MPICH se denominó MPICH1 y esta implementa el estándar MPI-1.1. A partir del 2006, la implementación más reciente se llama MPICH2.

2.1.6 OpenMP

OpenMP [25] es un API de programación que permite añadir concurrencia a las aplicaciones mediante paralelismo de memoria compartida. Se basa en la creación de hilos de ejecución paralelos compartiendo las variables del proceso padre que los crea.

Está disponible en múltiples plataformas y lenguajes, desde las derivadas de UNIX hasta la plataforma Windows. Existen extensiones para los lenguajes más conocidos como “C”, “C++”

OpenMP se basa en el modelo fork-join, paradigma que proviene de los sistemas UNIX, donde una tarea muy pesada se divide en K hilos (fork) con menor peso, para luego "recolectar" sus resultados al final y unirlos en un solo resultado (join).

Cuando se incluye una directiva OpenMP dentro de un código secuencial, implica que se incluye una sincronización obligatoria en todo el bloque. Es decir, el bloque de código se marcará como paralelo y se lanzarán hilos según las características que nos dé la directiva, y al final de ella habrá una barrera para la sincronización de los diferentes hilos (salvo que implícitamente se indique lo contrario con la directiva nowait). Este tipo de ejecución se denomina fork-join.

2.2 Virtualización con Xen

Xen [26] es un software de código abierto que permite desplegar máquinas virtuales. Su finalidad es la ejecución de distintos sistemas operativos a partir de cualquier hardware y, para ello, utiliza una técnica llamada paravirtualización [27] que permite incrementar el nivel de rendimiento con respecto a los resultados que ofrecen las técnicas tradicionales de virtualización. Estudios realizados sobre el uso de *Xen* manifiestan que la penalización sufrida por las máquinas virtuales es de un 2% y en casos extremos de un 8% respecto a la ejecución de forma nativa.

La principal desventaja de la paravirtualización, fórmula en la que se basa XEN para realizar la virtualización, es que el sistema instalado en las máquinas virtuales debe ser modificado para su perfecta ejecución.

Pero en la actualidad con los avances tecnológicos en los procesadores y la aparición de las instrucciones de virtualización (Intel-VT [28] y AMD-V [29]), ya no es necesaria la modificación del sistema operativo huésped para ser ejecutado como máquina virtual.

Desde la concepción del prototipo inicial de *Xen* en 2003 se estudió la posibilidad de mover de máquinas virtuales desde un ordenador físico a otro. Hasta la versión 3.0 esta migración era con pérdidas de servicios, pero a partir de esta versión se realiza sin tener que apagar la máquina como había sido hasta entonces. Esta migración hace que los servicios arrancados dentro del huésped no dejen de estar operativos maximizando el tiempo de utilización de la máquina virtual y los servicios ofrecidos por ella.

Esta característica permite manejar y distribuir la carga de cada hospedador de máquinas virtuales pudiendo crear servidores de alta disponibilidad.

Xen proporciona escalabilidad a las organizaciones debido a que el único requisito es la compra de otro servidor e instalar el software de alojamiento. Todas las máquinas virtuales creadas hasta la compra del nuevo servidor pueden ser utilizadas tanto en los servidores viejos como en los nuevos. Esta acción facilita el crecimiento descargando al administrador de tediosas áreas de gestión y configuración.

2.3 Almacenamiento compartido y distribuido

En este apartado se exponen los diferentes sistemas de ficheros y aplicaciones que pueden ser utilizados para compartir dispositivos y datos entre varios equipos.

2.3.1 NFS

NFS [30] (*Network File System*) fue diseñado originalmente por Sun Microsystems en 1985. NFS permite el acceso transparente a ficheros y directorios situados en máquinas remotas, para poder utilizarlos como si fueran locales. Sigue un esquema cliente-servidor, de forma que el nodo remoto se denomina servidor NFS y el nodo local se denomina cliente NFS. Ambos se comunican mediante RPC [31] ("*Remote procedure call*" o "llamadas a procedimientos remotos").

Características:

El sistema NFS está dividido principalmente en dos partes principales: un servidor y uno o más clientes. Los clientes acceden de forma remota a los datos que se encuentran almacenados en el servidor.

Las estaciones de trabajo locales utilizan menos espacio de disco debido a que los datos se encuentran centralizados en un único lugar pero pueden ser accedidos y modificados por varios usuarios, de tal forma que no es necesario replicar la información.

Los usuarios no necesitan disponer de un directorio "home" en cada una de las máquinas de la organización. Los directorios "home" pueden crearse en el servidor de NFS para posteriormente poder acceder a ellos desde cualquier máquina a través de la infraestructura de red.

También se pueden compartir a través de la red dispositivos de almacenamiento como disqueteras, CD-ROM o discos duros portátiles. Compartir estos dispositivos puede reducir la inversión en dichos dispositivos y mejorar el aprovechamiento del hardware existente en la organización.

Todas las operaciones sobre ficheros son síncronas, lo que significa que la operación sólo retorna cuando el servidor ha completado todo el trabajo asociado para

esa operación. En caso de una solicitud de escritura, el servidor escribirá físicamente los datos en el disco, y si es necesario, actualizará la estructura de directorios, antes de devolver una respuesta al cliente, con lo que se garantiza de esta forma la integridad de los ficheros.

2.3.2 OCFS2

Oracle Cluster File System [32] (OCFS) es un sistema de ficheros compartido, compatible con POSIX, desarrollado principalmente para los cluster. Es capaz de ofrecer a los equipos que lo utilizan tanto alto rendimiento como alta disponibilidad.

Las aplicaciones del cluster que son capaces de soportar paralelismo de E/S podrán aumentar su funcionalidad. En cambio las aplicaciones que no soporten los beneficios del paralelismo podrán disponer de un sistema de ficheros tolerante a fallos para incrementar la disponibilidad de uso de la aplicación.

Además de ser utilizado habitualmente con la herramienta de bases de datos *Real Aplication Cluster de Oracle* [33] (RAC), es utilizado también para proporcionar escalabilidad a servidores Web y servidores de ficheros, además de ofrecer tolerancia a fallos en servidores y como almacenamiento de imágenes de máquinas virtuales.

Algunas de las características más destacables de este sistema de ficheros son:

- Tamaños de bloque variables.
- Asignaciones flexibles a la hora de organizar los datos.
- Journaling (almacenamiento de los registros de transacciones de datos, para poder restablecerlos en caso de error).

2.3.2.1 iSCSI

Internet SCSI [34] (*Small Computer System Interface*) es un estándar oficial ratificado el 11 de Febrero de 2003 por la *Internet Engineering Task Force* [35] (IETF).

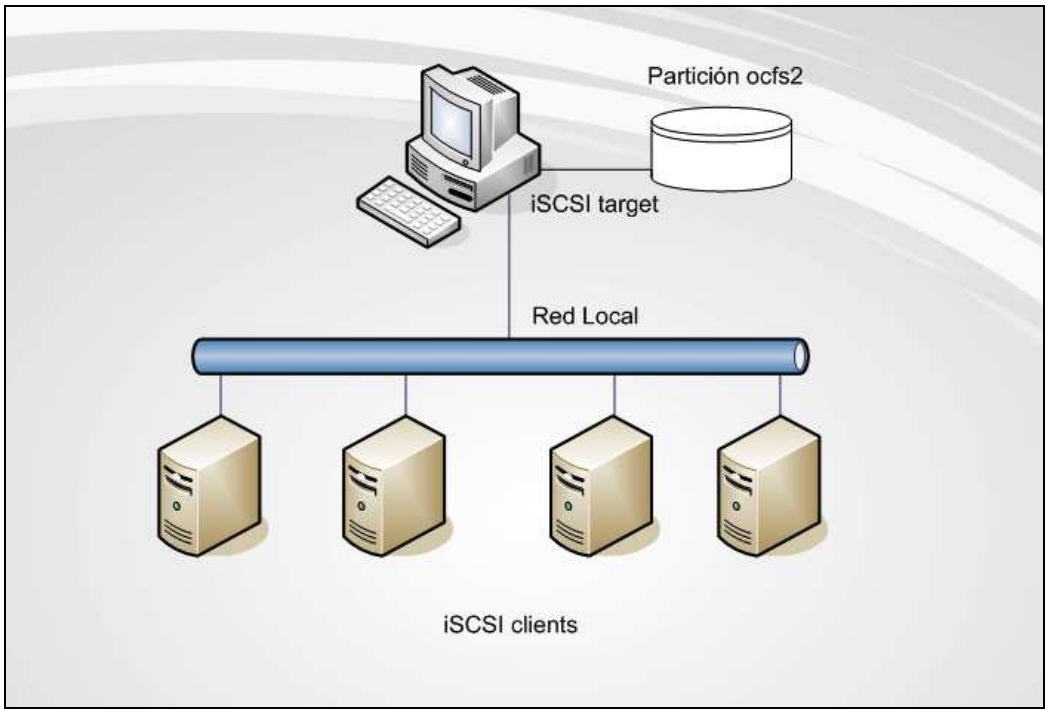

Figura 2: Arquitectura básica de iSCSI

El protocolo iSCSI utiliza TCP/IP para sus transferencias de datos. Al contrario que otros protocolos de red diseñados para almacenamiento que requieren de alta tecnología, como por ejemplo el canal de fibra (que es la base de la mayor parte de las redes de áreas de almacenamiento), solamente necesita una simple interfaz Ethernet (o cualquier otra red compatible TCP/IP) para funcionar. Esta facilidad de implementación permite generar una solución de almacenamiento centralizada de bajo coste sin la necesidad de realizar inversiones costosas ni sufrir las habituales incompatibilidades asociadas a las soluciones canal de fibra para redes de área de almacenamiento.

Los críticos de iSCSI argumentan que este protocolo tiene un peor rendimiento que el canal de fibra ya que se ve afectado por la sobrecarga que generan las transmisiones TCP/IP (cabeceras de paquetes, por ejemplo). Sin embargo las pruebas que se han realizado muestran un buen rendimiento de las soluciones iSCSI cuando se utilizan enlaces Gigabit Ethernet.

Dispositivos de almacenamiento

En el contexto de almacenamiento, iSCSI permite a un ordenador utilizar un iniciador iSCSI (*initiator*) para conectar a un dispositivo SCSI (*target*) como puede ser un disco duro o una cabina de cintas en una red IP para acceder a los mismos a nivel de bloque. Desde el punto de vista de los *drivers* y las aplicaciones de software, los

dispositivos parecen estar conectados realmente como dispositivos SCSI locales. Los entornos más complejos, consistentes en múltiples hosts y/o dispositivos son llamados redes de área de almacenamiento.

Permitir que múltiples equipos tengan acceso simultáneo a un dispositivo único es una tarea difícil pero muy común en los dispositivos SCSI. Sin comunicación equipo a equipo, cada uno de los nodos desconoce cuáles son las intenciones del resto de los equipos.

2.3.3 Lustre

Lustre [36] es un sistema de archivos distribuido *Open Source* (de código abierto), normalmente utilizado en clusters a gran escala. El proyecto intenta proporcionar un sistema de archivos para clusters de decenas de miles de nodos con petabytes de capacidad de almacenamiento, sin comprometer la velocidad o la seguridad, y está disponible bajo la Licencia GNU GPL.

Cluster File Systems son los diseñadores, desarrolladores y se encargan del mantenimiento de Lustre con colaboraciones de otras compañías y particulares.

Muchos de los superordenadores más rápidos del mundo son clusters que utilizan el sistema de archivos Lustre como herramienta de almacenamiento.

Lustre ha sido seleccionado por agencias y organismos de prestigio, incluyendo, en los Estados Unidos, los Departamentos de Energía y Defensa, la Fundación Nacional de la Ciencia, la agencia espacial NASA, la Oficina Nacional de Administración Oceánica y Atmosférica, además del Comisariado de Energía Atómica de Francia, el Centro Nacional de Supercomputación de Suiza y el Instituto Avanzado de Ciencia y Tecnología del Japón. Además, también es utilizado por empresas líderes en el mercado relacionadas con la biología, el video digital, el petróleo y gas, la industria aeroespacial, manufacturera y con otros entornos HPC comerciales.

El sistema de archivos Lustre es una compleja solución de software y los usuarios tienen la libertad de desplegarlo en el hardware y en el sistema de almacenamiento que deseen.

El sistema de ficheros considera a cada archivo almacenado en el sistema de archivos Lustre un objeto. Lustre presenta a todos los clientes una semántica POSIX [37] estándar y acceso concurrente a lectura y escritura para los objetos compartidos. Un sistema de archivos Lustre tiene cuatro unidades funcionales. Estas son: "*Meta Data Server*" (MDS) para almacenar los metadatos; un "*Object Storage Target*" (OST) para guardar los datos reales; un "*Object Storage Server*" (OSS) para manejar los OSTs; cliente(s) para acceder y utilizar los datos. Los OSTs son dispositivos de bloques. Un MDS, OSS, y un OST pueden residir en el mismo nodo o en nodos diferentes. Lustre no administra directamente los OSTs, y delega esta responsabilidad en los OSSs para asegurar la escalabilidad para grandes clusters y supercomputadores.

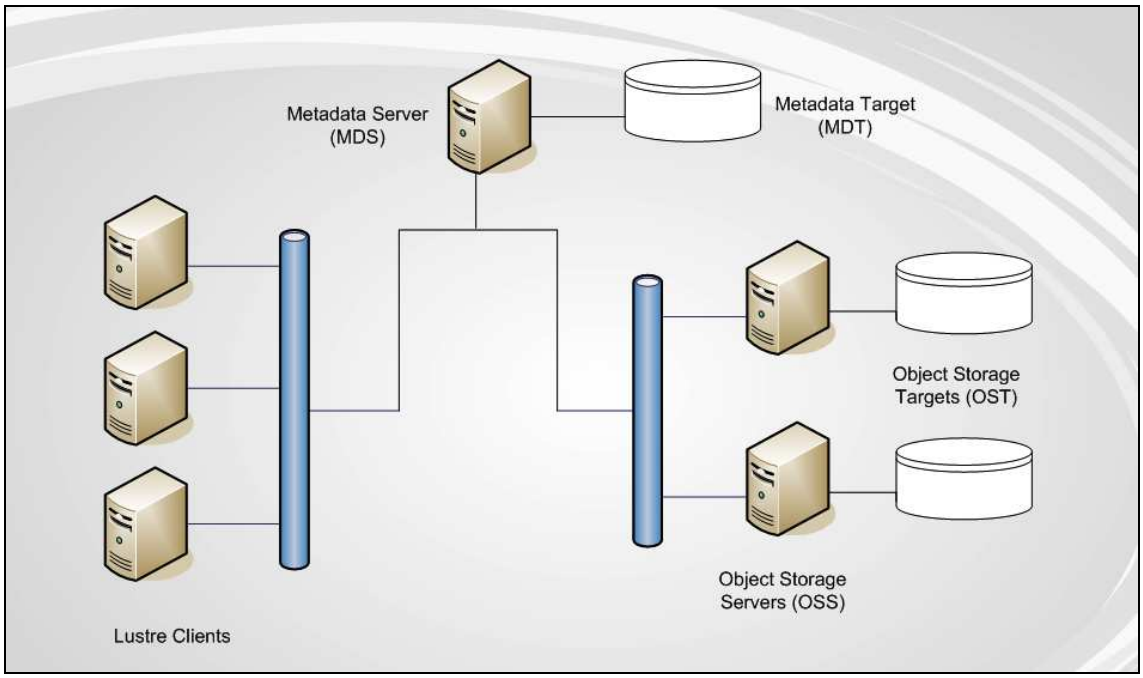

Figura 3: Arquitectura de Lustre

En un "*Massively Parallell Processor*" (MPP), los procesadores pueden acceder al sistema de archivos Lustre redirigiendo sus peticiones de EI/S hacia el nodo con el servicio lanzador de tareas, si está configurado como un cliente Lustre. Aunque es el método más sencillo, en general proporciona un bajo rendimiento. Una manera ligeramente más complicada de proporcionar un rendimiento global muy bueno consiste en utilizar la biblioteca "liblustre". "Liblustre" es una biblioteca de nivel de usuario que permite a los procesadores montar y utilizar el sistema de archivos Lustre como un cliente, sorteando la redirección hacia el nodo de servicio. Utilizando "liblustre", los procesadores pueden acceder al sistema de archivos Lustre, incluso si el nodo de servicio en el que se lanzó el trabajo no es un cliente Lustre. "Liblustre" proporciona un

mecanismo para mover datos directamente entre el espacio de aplicación y los OSSs de Lustre sin necesidad de realizar una copia de datos a través del núcleo ligero, logrando así una baja latencia, y gran ancho de banda en el acceso directo de los procesadores al sistema de archivos Lustre.

Gracias a su gran rendimiento y escalabilidad características, utilizar Lustre en sistemas MPP es lo más adecuado. "Liblustre" es la mayor diferencia de diseño entre Lustre en un MPP y los clusters convencionales.

2.3.4 PVFS

El nombre proviene de las siglas de "*Parallel Virtual File System*" [38] y es un sistema de ficheros paralelo que proporciona alta eficiencia y escalabilidad. Su diseño se centra en clusters Linux, en los que proporciona un gran ancho de banda en las operaciones de lectura y escritura concurrentes realizadas desde múltiples procesos o hilos a un fichero común. Se distribuye como software libre y no requiere de ningún hardware especial para que funcione.

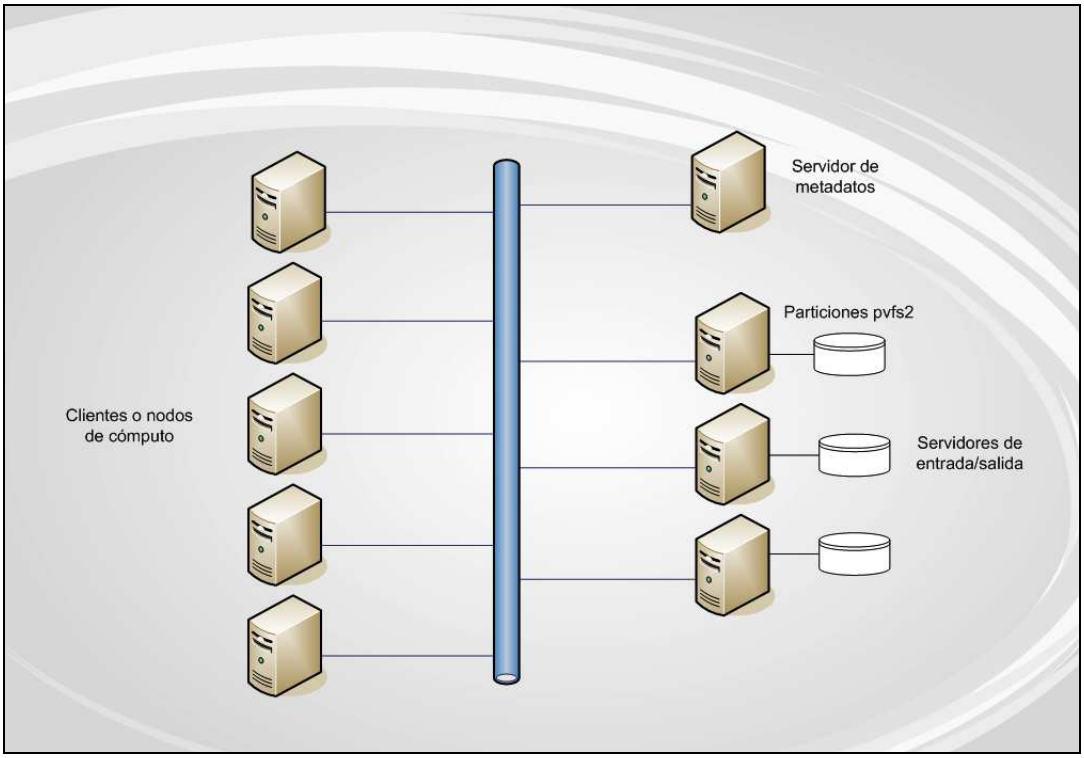

Figura 4: Arquitectura básica de PVFS

Para conseguir un alto rendimiento o un gran ancho de banda las operaciones de lectura y escritura concurrentes, PVFS distribuye los datos en múltiples nodos del

cluster, denominados *I/O nodes* o nodos de E/S. Distribuyendo los datos en múltiples nodos, los clientes poseen diferentes rutas hacia los datos, eliminado de esta forma los cuellos de botella y mejorando el ancho de banda para múltiples clientes.

Características principales:

- Transparencia para el usuario: permite prescindir de las llamadas al kernel en los accesos al sistema de ficheros, gracias al uso de una API nativa. Ésta implementa un subconjunto de operaciones UNIX que permiten contactar directamente con los servidores PVFS utilizando los comandos habituales (*ls, cd*, etc.).

- Distribución física de los datos a través de múltiples discos en distintos nodos.

- Se monta en todos los nodos y en el mismo directorio simultáneamente, permitiendo el acceso concurrente a todos los ficheros del sistema PVFS a través del mismo esquema de directorios.

- Fácil instalación.

- Posee múltiples interfaces, incluyendo MPI I/O.

Componentes:

- Servidor de metadatos ("*mgr*"): gestiona los metadatos de todos los ficheros.

- Servidor de E/S ("*iod*"): gestiona el almacenamiento y recuperación de los datos almacenados en el disco local del nodo

- API nativa de PVFS ("*libpvfs*"): maneja las operaciones necesarias para mover datos entre las caché de usuarios y los servidores PVFS, manteniendo las operaciones transparentes a los primeros.

- Soporte para el kernel: provee la funcionalidad para montar sistemas PVFS en los nodos Linux lo que permite a los programas existentes acceder a los datos almacenados en PVFS sin modificaciones.

2.4 Benchmarking

El *benchmarking* [39] es una técnica utilizada para medir el rendimiento de un sistema o componente de un sistema. La palabra *benchmark* es un anglicismo traducible al castellano como comparativa. Si bien también puede encontrarse esta palabra haciendo referencia al significado original en la lengua anglosajona, es en el campo informático donde su uso está más ampliamente extendido. Más formalmente puede entenderse que un benchmark es el resultado de la ejecución de un programa informático o un conjunto de programas en una máquina, con el objetivo de estimar el rendimiento de un elemento concreto o la totalidad de la misma, y poder comparar los resultados con máquinas similares. En términos de ordenadores, un benchmark podría ser realizado en cualquiera de sus componentes, ya sea CPU, RAM, tarjeta gráfica, etc. También puede ser dirigido específicamente a una función dentro de un componente, por ejemplo, la unidad de coma flotante de la CPU; o incluso a otros programas.

La tarea de ejecutar un benchmark originalmente se reducía a estimar el tiempo de proceso que lleva la ejecución de un programa (medida por lo general en miles o millones de operaciones por segundo). Con el correr del tiempo, la mejora en los compiladores y la gran variedad de arquitecturas y situaciones existentes convirtieron a esta técnica en toda una especialidad. La elección de las condiciones bajo la cual dos sistemas distintos pueden compararse entre sí es especialmente ardua, y la publicación de los resultados suele ser objeto de candentes debates cuando éstos se abren a la comunidad.

El *Benchmarking* es también un proceso continuo de mejora de productos y servicios, con el fin realizar nuevos productos más competitivos frente a competidores más duros o aquellas compañías reconocidas como líderes en la industria.

Características

Los benchmark tienen las siguientes funcionalidades:

- Comprobar si las especificaciones de los componentes están dentro del margen propio del mismo.
- Maximizar el rendimiento con un presupuesto dado.

- Minimizar costes manteniendo un nivel mínimo de rendimiento.
- Obtener la mejor relación costo/beneficio (con un presupuesto o unas exigencias dadas).
- Ayuda a lograr una posición más competitiva.

2.4.1 Tipos de benchmarks

Según el rendimiento objetivo, existen diferentes tipos de benchmark:

Sintéticos vs Aplicaciones

Sintéticos: están especialmente diseñadas para medir el rendimiento de un componente individual de un ordenador, normalmente llevando el componente escogido a su máxima capacidad.

Aplicaciones: herramientas basadas en aplicaciones reales, simulan una carga de trabajo para medir el comportamiento global del equipo.

Bajo nivel vs Alto nivel

Test de Bajo nivel:

Miden directamente el rendimiento de los componentes Ejemplo: el reloj de la CPU, los tiempos de la DRAM y de la caché SRAM, tiempo de acceso medio al disco duro, latencia, tiempo de cambio de pista, etc.

Test de Alto nivel:

Están más enfocados a medir el rendimiento de la combinación componente/controlador/SO de un aspecto específico del sistema, como por ejemplo el rendimiento de E/S con ficheros, o el rendimiento de una determinada combinación de componentes/controlador/SO/aplicación.

2.4.2 NAS BT-IO

NAS BT-IO, es una extensión del benchmark NAS BT [40]. Este test simula el patrón de acceso de E/S de BT (*Block Tri-Diagonal*). La estructura de datos de la prueba consiste en series tridimensionales. El test realiza escrituras puntuales sobre el mismo fichero. El test alterna etapas de cómputo con etapas de E/S. Inicialmente, todos los nodos de cómputo abren el fichero de forma colectiva y declaran una vista sobre varias regiones del fichero. Cada cinco iteraciones de cómputo, cada proceso escribe la solución al fichero a través de una operación de escritura colectiva. Al finalizar la ejecución, el fichero con los resultados es leído de forma colectiva y la solución obtenida es verificada. El test permite ejecutar distintos tamaños de problemas. Este test, junto a FLASH I/O [41], son los test más usados en la comunidad científica para evaluar soluciones de E/S paralela.

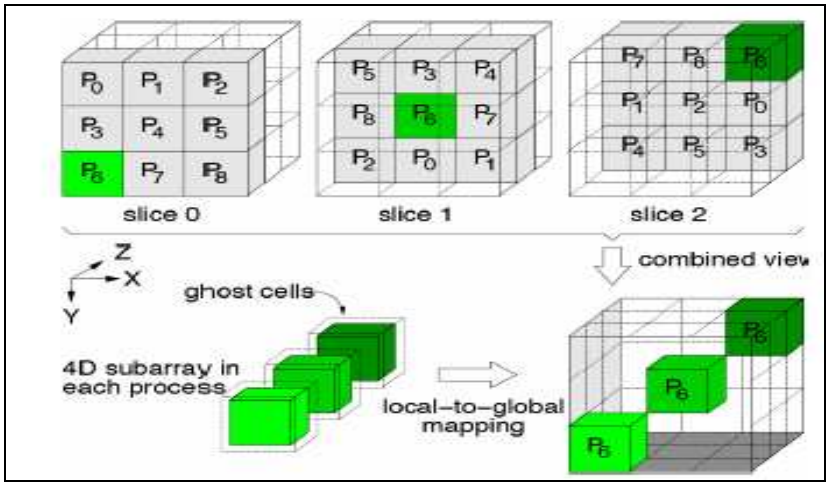

Figura 5: Estructura BT-IO

2.4.3 COLLPERF

Coll_perf [42] es un benchmark integrado dentro de ROMIO [43], utilizado para estimar el funcionamiento de rutinas de E/S colectivas de MPI-IO . La prueba mide el ancho de banda de E/S tanto para la lectura como para la escritura de un archivo, con patrón acceso al archivo de una serie tridimensional distribuida por bloques.

Para la E/S colectiva de ROMIO, todos los nodos computan, y el tamaño del buffer colectivo es 16MBytes (por defecto). La división de datos se realiza por la asignación de un número de procesos sobre cada dimensión cartesiana

2.4.4 Linpack

Linpack [44] es el benchmark más popular a la hora de evaluar clusters y supercomputadoras. La característica principal de Linpack es que hace un uso muy intensivo de las operaciones de punto flotante, por lo que sus resultados son muy dependientes de la capacidad de la Unidad de Punto Flotante (FPU) del sistema. Además pasa la mayor parte del tiempo ejecutando unas rutinas llamadas BLAS [45] ("*Basic Linear Algebra Subroutines*" o Subrutinas de Álgebra Lineal Básica). Como hay dos tipos de estas librerías (una codificada en ensamblador y otra en Fortran), el resultado también dependerá mucho de esto. El principal sistema de ejecución que utiliza este benchmark se denomina DAXPY y se encuentra en la librería BLAS. Su ejecución consume cerca del 90% del tiempo que es utilizado para pasar este benchmark. DAXPY realiza el siguiente cálculo: *"y(i) := y(i) + a * x(i)"*

Es por esta razón que en realidad se puede decir que lo que mide Linpack es la velocidad del sistema para DAXPY.

Por otra parte, al realizar esencialmente cálculos con matrices es un test fácilmente paralelizable, y se puede utilizar para medir la eficiencia de sistemas multiprocesador

Principios matemáticos:

El Linpack Benchmark trata de resolver un problema de matrices generales densas *"Ax = b"* a tres niveles de tamaño y con oportunidad de optimización: problemas de 100 x 100 (optimización del bucle interno), problemas de 1.000 por 1.000 (optimización de tres bucles - el programa entero) y un problema paralelo escalable. Estas matrices son creadas usando un generador de números pseudoaleatorios, pero forzando los números para que pueda ejecutarse un pivoteo parcial con **Eliminación Gaussiana**. Ejecuta dos rutinas básicas: una que descompone la matriz, y otra que resuelve el sistema de ecuaciones basándose en la descomposición de la primera matriz. Para una matriz de N x N la primera rutina lleva N^3 operaciones de coma flotante, mientras que la segunda ejecuta N^2 operaciones de este tipo.

3 ANÁLISIS

3.1 Visión general del proyecto

Para el desarrollo de este proyecto es fundamental contar con una herramienta de gestión y copia de imágenes para facilitar el manejo de los nodos. Para este propósito se barajaron diversas posibilidades disponibles en el mercado, como Norton Ghost, pero se buscaba abaratar al máximo el despliegue de este cluster, por tanto esta idea se desechó de forma inmediata. Se buscaron otras alternativas de tipo GPL, y se dio con la herramienta que se consideró definitiva, el Linux Cluster Manager. Esta aplicación ofrece todas las funcionalidades necesarias para gestionar imágenes y gestionar los nodos de un cluster, desde un servidor y en modo gráfico.

Uno de los objetivos planteados en este proyecto es poder ofrecer un cluster de alto rendimiento para la investigación y la docencia en la Universidad Carlos III de Madrid, y por consiguiente, es necesario que el alumnado y el profesorado tengan libre acceso a este cluster. Para satisfacer este requisito, se ha elegido la aplicación NIS, gracias a la cual todos los alumnos y profesores que tengan cuenta en las aulas de Linux

del Laboratorio del Departamento de Informática (LDI) podrán acceder de forma remota con sus cuentas al cluster mediante aplicaciones del estilo del *Putty* o *ssh*.

Para el lanzamiento de trabajos se han estudiado diferentes alternativas disponibles de código abierto, pero debido a las facilidades de instalación y uso, y a la experiencia que se ha adquirido anteriormente en este tipo de herramientas, se han seleccionado dos sobre el resto, *Torque* y *Condor*. En la sección 2.1.4 se comentaron brevemente las características y capacidades de cada uno de ellos antes de decantarse por uno o por otro.

Uno de los requisitos iniciales de este proyecto, es ofrecer un entorno de trabajo para los alumnos de la asignatura de Arquitectura de Computadores II para poder lanzar sus prácticas y trabajos, y las aplicaciones que se utilizan para lanzar los trabajos son tanto MPICH, como OpenMP, y por tanto deben ser incluidas desde el principio como requisitos básico de instalación en el cluster.

Una de las funcionalidades que se ha decidido añadir a los nodos del cluster es que puedan tener la posibilidad de lanzar máquinas virtuales, y para este cometido se ha decidido implantar en todos ellos un kernel de Xen, debido a la facilidad de manejo y a la experiencia que se tiene previamente en el LDI en este tipo de tecnología. Cabe recordar que lo servidores de Linux que actualmente están dando soporte a las aulas y a los servicios básicos del Laboratorio, son realmente máquinas virtuales sobre Xen.

3.2 Estudio de las posibles arquitecturas

Los recursos que estaban disponibles desde el inicio del proyecto son entre 32 y 34 ordenadores (anteriormente utilizados como equipos informáticos de las aulas del Laboratorio del Departamento), dos switch de 24 entradas cada uno de ellos, y un equipo con buenas prestaciones, que era utilizado como servidor del Laboratorio.

Los equipos de las aulas disponen de un procesador AMD K7, a 1700 Mhz y con 1GB de memoria Ram cada uno de ellos, mientras que el servidor contaba con un procesador Pentium III, Dual de 1Ghz y 2GB de memoria.

De los dos switch citados anteriormente, el primero de ellos cuenta con 1 gigabit de velocidad mientras que el segundo trabaja a 100 megabits por segundo.

En vista a los recursos disponibles inicialmente, se sopesaron varias arquitecturas dentro de todas las posibles combinaciones que se pudieran hacer, pero finalmente se tomó una decisión sobre las 2 arquitecturas más realistas.

La primera de ellas tenía como ventaja que el cluster entero de unos 23 nodos, trabajase a velocidad de gigabit. La desventaja principal era que dejaban de usarse más de 10 equipos en la arquitectura y un switch, aunque de menor velocidad, debido a que el switch de esta velocidad solo contaba con 24 entradas.

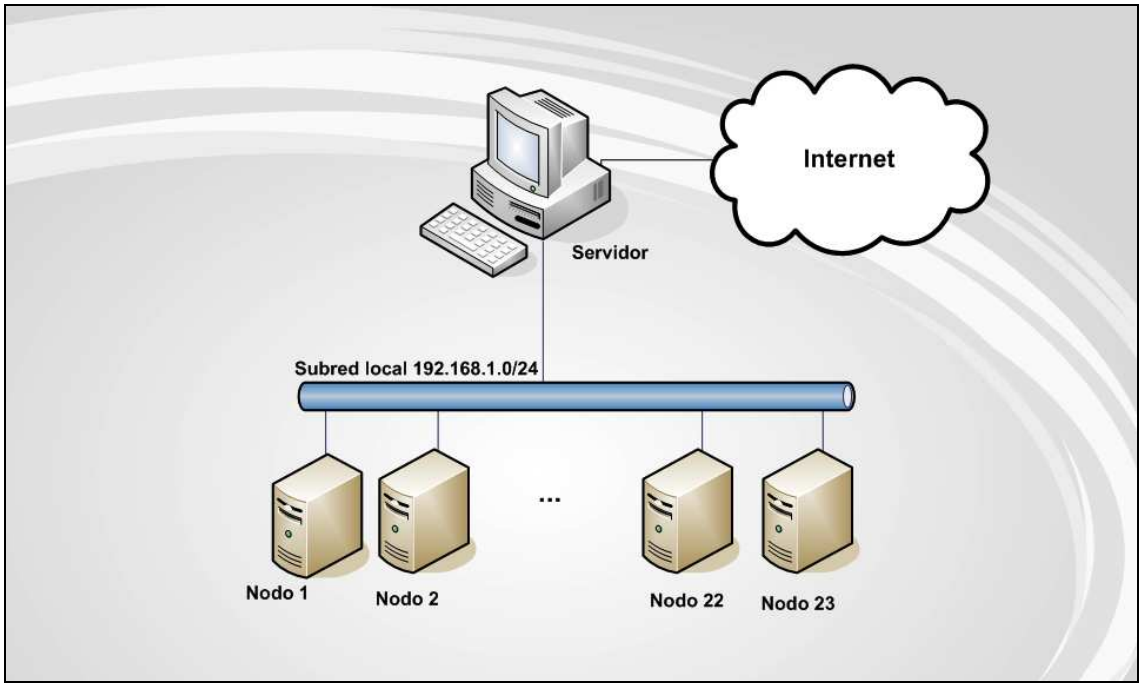

Figura 6: Primer prototipo de arquitectura

La otra arquitectura que se ideó como posible despliegue del cluster, constaba de un servidor, dos switch con las velocidades de datos diferentes, y 32 ordenadores para tener 2 equipos de respaldo en caso de que cualquiera de los nodos se viese comprometido y no pudiese seguir funcionando.

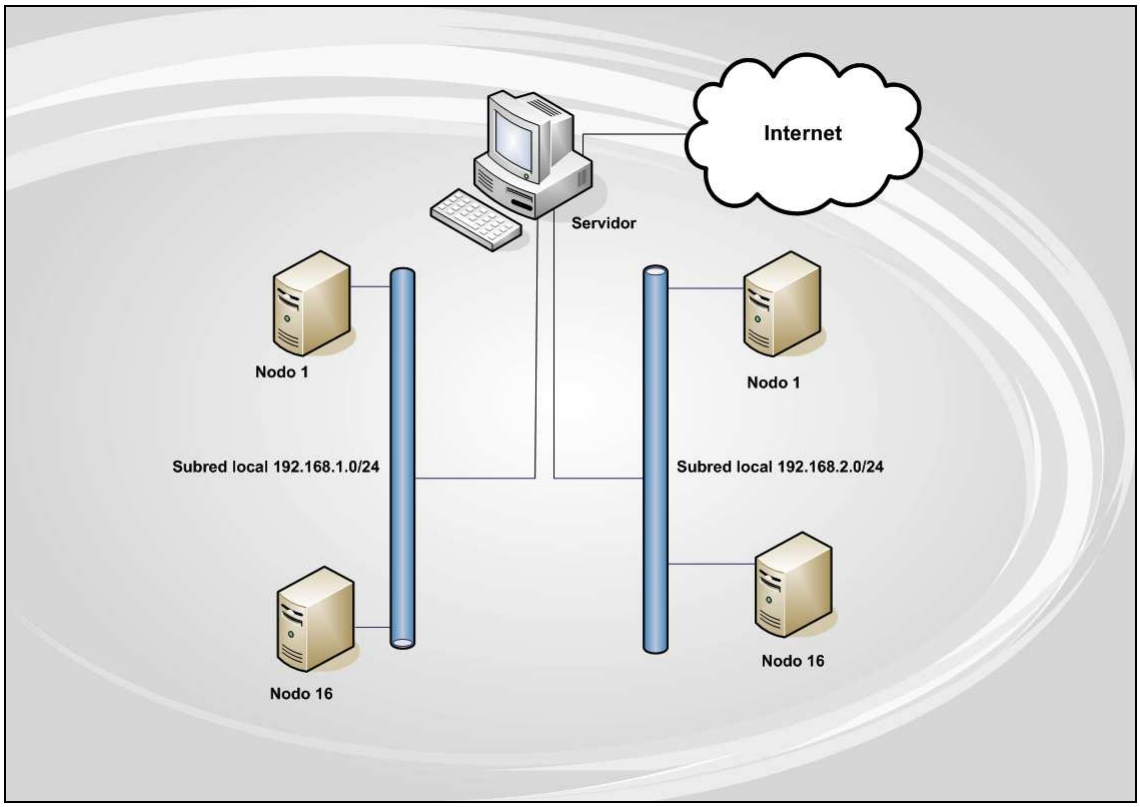

Figura 7: Segundo prototipo de arquitectura

Esta segunda arquitectura fue mejor recibida ya que no desperdiciaban ninguno de los recursos de los que se disponían y permitía realizar un mayor abanico de pruebas y configuraciones.

3.3 Especificación de requisitos de usuario

El objetivo de la especificación es definir de manera clara y precisa todas las funcionalidades y restricciones del sistema que se desea construir o desplegar.

Los requisitos deberán ser validados por todos los implicados antes de llevarse a cabo la implementación del sistema. Se presentarán las capacidades del sistema, las restricciones del mismo, es decir, las características que deben cumplir los usuarios y el entorno operacional sobre el que trabajará.

A continuación, se especifica el significado de cada uno de los campos que contiene la tabla de cada requisito:

Identificador: código unívoco que identifica a cada requisito de usuario. Las siglas RU indican que es un requisito de Usuario, y XX el número de requisito comenzando en el 01.

Prioridad: se refiere a la necesidad de implementar unos requisitos antes que otros.

Necesidad: indica la negociación de la realización del requisito entre nosotros y el cliente:

Esencial: no es posible la negociación, ya que es imprescindible.

Deseable: el cliente desea que se realice lo pedido.

Opcional: la implementación del mismo es opcional.

Claridad: indica la ambigüedad de un requisito, es decir, si está sujeto a diferentes interpretaciones.

Verificabilidad: indica si es posible probar que el software aplica el requisito.

Descripción: se describe al requisito de una forma clara y concisa.

3.4 Requisitos de capacidad

Los requisitos de capacidad indican las funcionalidades que va a ofrecer el desarrollo de esta arquitectura.

Identificador: RU-C-01		
Prioridad: ☒Alta ☐ Media ☐ Baja		Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional		
Claridad: ☒ Alta ☐ Media ☐ Baja		Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable	
Descripción:	Estará disponible el acceso al cluster desde el exterior, mediante SSH.	

Identificador: RU-C-02		
Prioridad: ☒Alta ☐ Media ☐ Baja		Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional		
Claridad: ☒ Alta ☐ Media ☐ Baja		Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable	
Descripción:	Se permitirá a los usuarios tener acceso a los datos de sus cuentas dentro del cluster.	

Identificador: RU-C-03		
Prioridad: ☒Alta ☐ Media ☐ Baja		Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional		
Claridad: ☒ Alta ☐ Media ☐ Baja		Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable	
Descripción:	Se permitirá a los usuarios enviar trabajos al cluster.	

Identificador: RU-C-04		
Prioridad: ☒Alta ☐ Media ☐ Baja		Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional		
Claridad: ☒ Alta ☐ Media ☐ Baja		Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable	
Descripción:	Se permitirá a los usuarios la ejecución de trabajos paralelos (MPI y openMP).	

Identificador: RU-C-05		
Prioridad: ☒Alta ☐ Media ☐ Baja		Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional		
Claridad: ☒ Alta ☐ Media ☐ Baja		Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable	
Descripción:	Se podrán crear y gestionar máquinas virtuales en los nodos del cluster.	

3.5 Requisitos de restricción

A continuación de enumeran los requisitos de restricción

Identificador: RU-R-01		
Prioridad: ☒Alta ☐ Media ☐ Baja		Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional		
Claridad: ☒ Alta ☐ Media ☐ Baja		Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable	
Descripción:	Se deberán configurar diferentes sistemas de ficheros, tanto distribuidos como locales en los diferentes cluster implementados.	

Identificador: RU-R-02	
Prioridad: ☒Alta ☐ Media ☐ Baja	Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☒ Alta ☐ Media ☐ Baja	Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable
Descripción:	Se deberá gestionar el envío de las imágenes de los nodos mediante una aplicación de Código Abierto desde el servidor.

Identificador: RU-R-03	
Prioridad: ☒Alta ☐ Media ☐ Baja	Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☒ Alta ☐ Media ☐ Baja	Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable
Descripción:	Se deberá gestionar la asignación de IPs de los nodos del cluster mediante un servidor de DHCP.

Identificador: RU-R-04	
Prioridad: ☒Alta ☐ Media ☐ Baja	Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☒ Alta ☐ Media ☐ Baja	Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable
Descripción:	Todos los equipos que formarán parte del cluster deberán ser equipos reutilizados.

Identificador: RU-R-05	
Prioridad: ☒Alta ☐ Media ☐ Baja	Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☒ Alta ☐ Media ☐ Baja	Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable
Descripción:	Los trabajos deberán ser lazados al cluster por la aplicación gestora.

Identificador: RU-R-06	
Prioridad: ☒Alta ☐ Media ☐ Baja	Fuente: Tutor
Necesidad: ☒ Esencial ☐ Deseable ☐ Opcional	
Claridad: ☒ Alta ☐ Media ☐ Baja	Verificabilidad: ☒ Alta ☐ Media ☐ Baja
Estabilidad:	Estable
Descripción:	El sistema operativo de todo el cluster deberá ser la nueva versión de Debian, Lenny.

4 DISEÑO DEL CLUSTER

4.1 Arquitectura del cluster

En este proyecto se ha considerado que la mejor forma de distribuir el cluster que va a ser desarrollado es en forma de cluster tipo *Beowulf*. Esta decisión se debe a que es la mejor forma de controlar un número relativamente reducido de nodos (se tienen 32 equipos divididos en dos subredes), y además la principal función de este cluster va a ser ejecutar los trabajos que van a ser enviados desde el servidor.

De esta forma se consigue también que se reduzca la salida de los nodos a Internet, abriendo esta opción solo en ocasiones especiales, tales como actualizaciones del sistema o instalaciones de alguna aplicación específica. La arquitectura del cluster propuesta queda reflejada de forma específica en la siguiente figura:

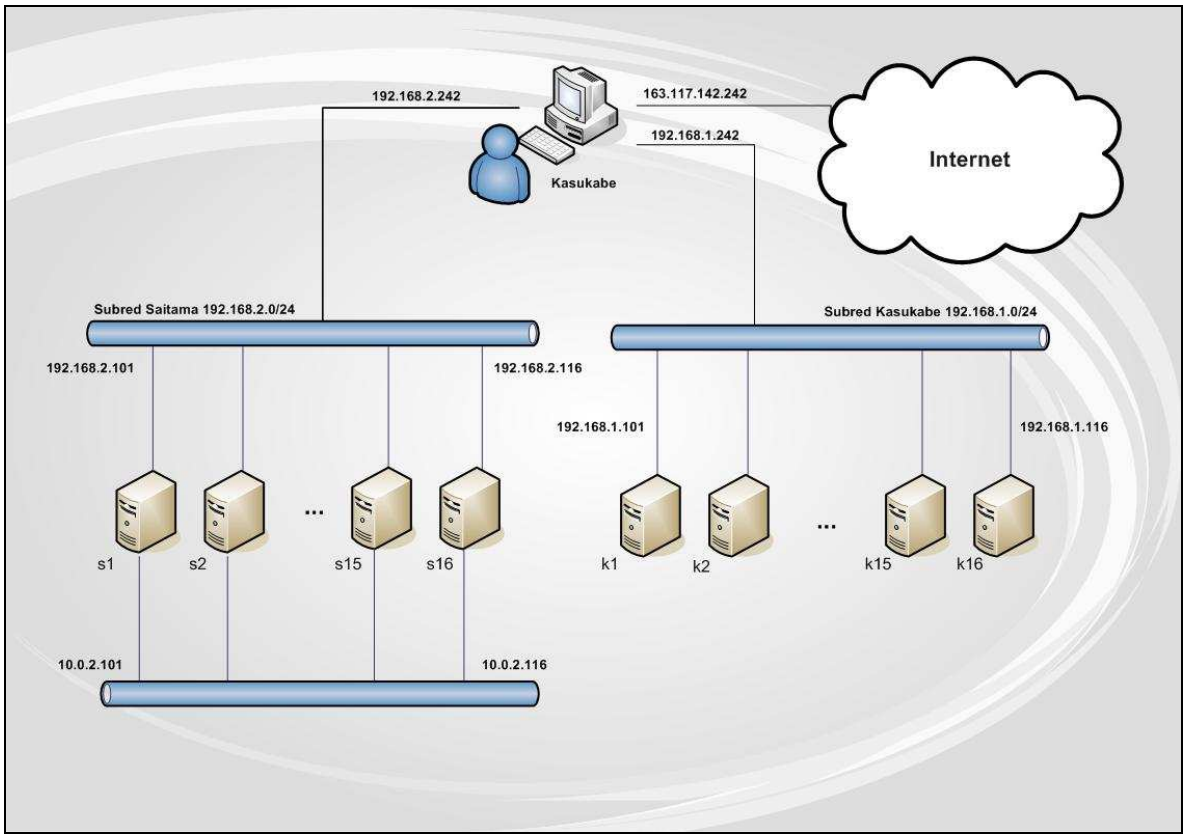

Figura 8: Arquitectura del cluster propuesto

El diseño inicial del cluster se compone de los siguientes elementos: un servidor (Pentium III, Dual 1 Ghz, 2GB de memoria), 32 nodos (AMD K7, 1700 Mhz, 1GB de memoria), dos switch de 24 puertos a velocidad de 100 Mbps y un switch más de 24 bocas a velocidad de Gigabit. Además se dispondrá de los siguientes periféricos: un monitor, un teclado y un ratón permanentes en el servidor, y otros dos monitores y dos teclados más a disposición de los nodos, por si en algún momento llegara a requerirse la interacción directa en alguno de ellos.

Se puede observar en la Figura 8 cómo el cluster va a estar subdividido en dos partes, que conforman dos clusters separados. El servidor, como se especifica en el diagrama, tiene tres tarjetas de red, una para la salida a Internet y otra para conectarse a cada uno de los dos subcluster que va a gestionar.

El primero de las subredes pasará a llamarse “kasukabe”, como el servidor, ya que fue el primero que se implemento y en principio iba a ser el único que se iba a desarrollar. Más tarde se incrementó el número de ordenadores disponibles y se consideró que en vez de tener una sola red con 32 nodos, se creara uno nuevo y así

contar con dos clusters separados de 16 nodos cada uno. De este modo nació la subred llamada “saitama”.

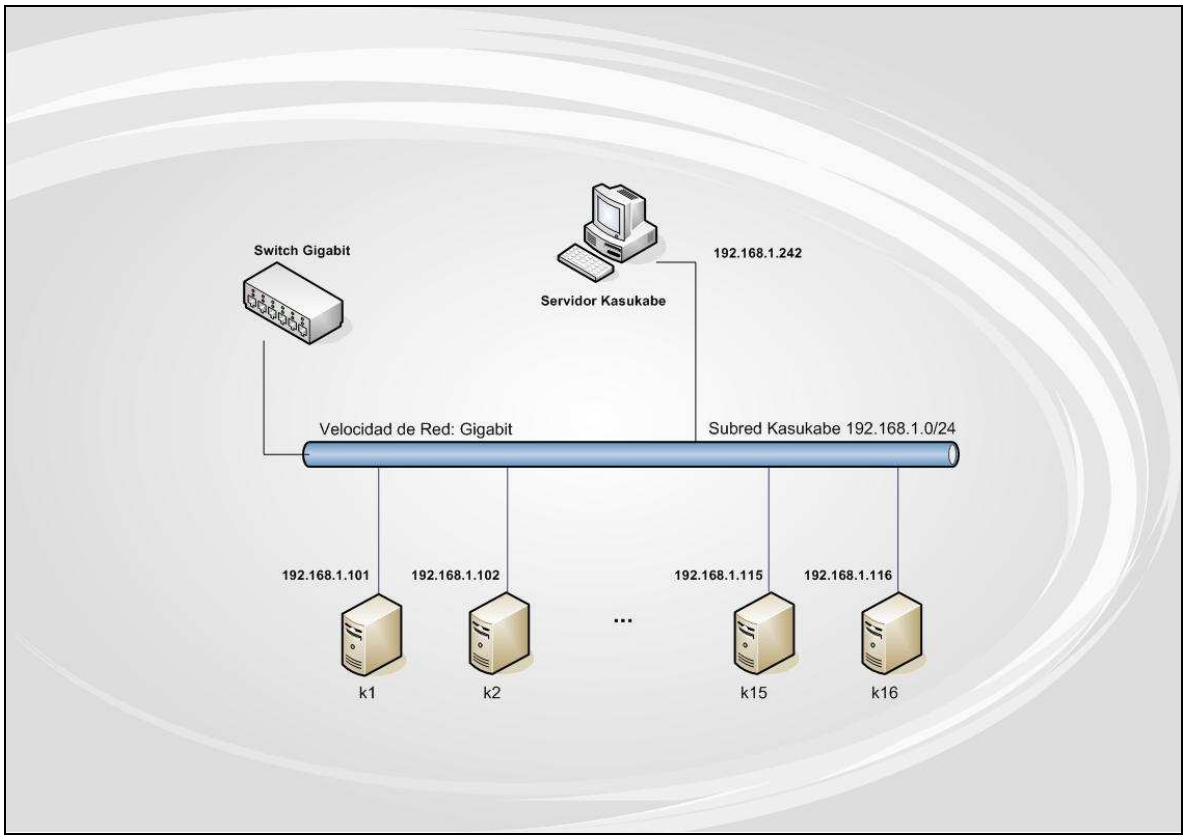

Figura 9: Arquitectura general de la subred kasukabe

Como se puede ver en la Figura 9, los nodos del cluster kasukabe cuentan con una sola conexión entre ellos y con el servidor, ya que solo disponen de una tarjeta de red. Sin embargo para realizar pruebas y hacer comprobaciones, se han instalado dos tarjetas de red en los nodos de la subred saitama, aunque las especificaciones de cada una de las redes se verán en el punto 4.1.1.

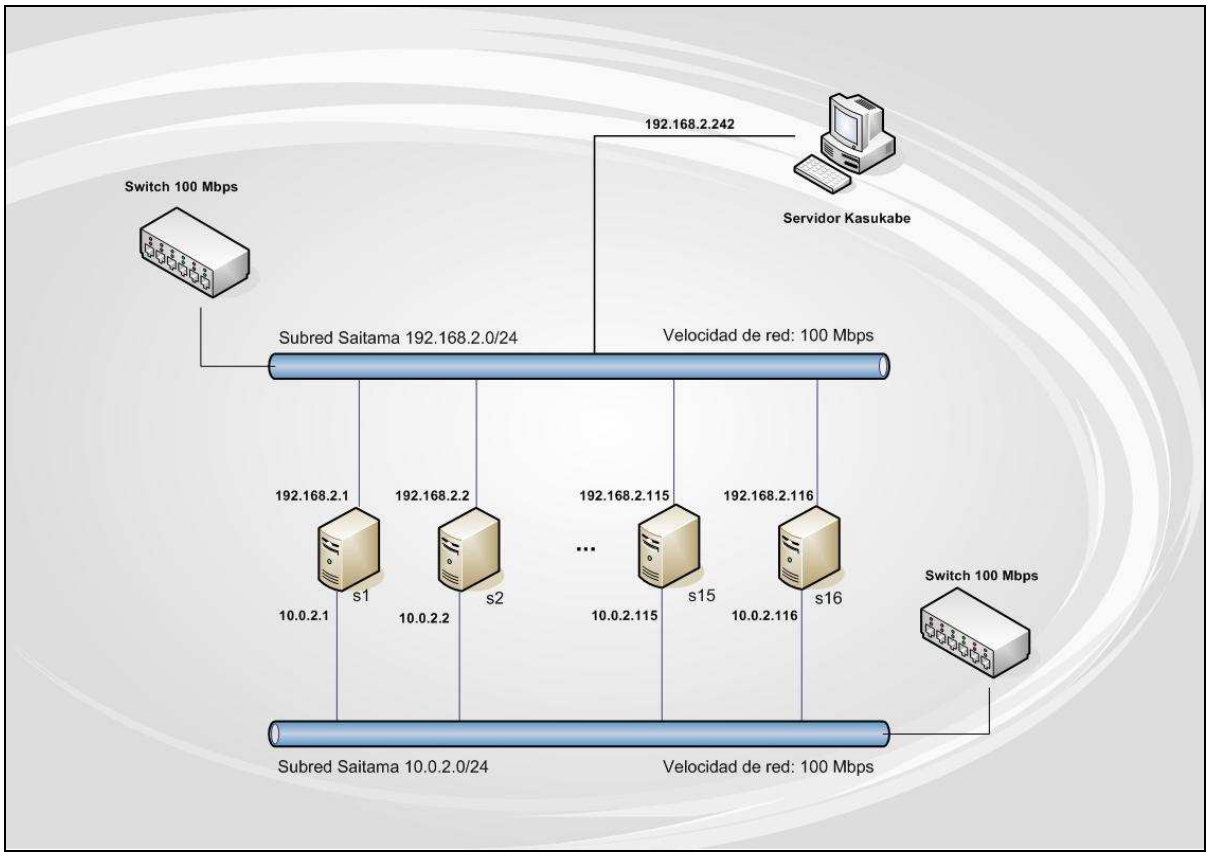

Figura 10: Arquitectura general de la subred saitama

A continuación se van a explicar de forma más específica las diferentes partes de las que se compone el cluster y sus diferentes funciones.

4.1.1 Redes

El rango de direcciones disponibles para cada uno de los cluster está limitado a 255 direcciones, pero son más que suficientes ya que en principio el número de nodos con los que va a contar cada cluster va a ser de 16. Aún así se han instalado en los equipos del cluster la herramienta de virtualización Xen, con lo que cada uno de los nodos del cluster podrá albergar una o varias máquinas virtuales. Estas máquinas se van a arrancar de forma que solo esté encendida una por equipo, y de esta forma tan solo se ampliará en numero de direcciones que se deben asignar a otras 16. Por tanto en total cada uno de los clusters tendrá reservados un total de 32 direcciones entre máquinas físicas y virtuales.

Como ya se ha comentado en el punto anterior, el servidor cuenta con tres tarjetas de red, una para la salida a Internet, otra para enlazarse con el cluster kasukabe y la última para enlazarse con el cluster saitama.

La dirección IP 163.117.142.242 se le ha asignado para la conexión a Internet, ya que es una de las direcciones de las que dispone el Laboratorio del Departamento de Informática y estaba libre. Esta conexión cuenta con una tarjeta de red con una velocidad de 100 Mbps, ya que en principio no se considera necesario que cuente con una conexión mejor.

La dirección IP 192.168.1.242 ha sido asignada para la conexión con el cluster kasukabe. Para toda la conexión del cluster se le ha asignado el rango de direcciones 192.168.1.0/24, y por tanto el servidor ha conservado la terminación que se le ha asignado en la dirección IP que sale al exterior.

La dirección IP 192.168.2.242 ha sido asignada para la conexión con el cluster saitama. Para toda la conexión del subcluster se le ha asignado el rango de direcciones 192.168.2.0/24, y como en el caso anterior el servidor conserva la terminación que se le ha asignado en la dirección IP que sale al exterior.

Como se puede observar los nodos de la subred kasukabe cuentan con una sola tarjeta de red que les conecta directamente entre ellos y con el servidor. La conexión en este subcluster se ha definido a velocidad de Gigabit, por lo que se ha necesitado que todos y cada uno de los equipos que forman parte de esta subred cuenten con una tarjeta de red de este tipo y una más que será la que conectará el servidor con ellos. Además para que la conexión sea completamente a nivel de Gigabit, se ha necesitado un switch de estas características que será al que se conecten todos ellos junto con el servidor.

A los nodos de la subred kasukabe se les había asignado inicialmente el nombre de kasukabe-X siendo la X el número entre 1 y 16 correspondiente a la posición que ocupan. Pero al tener que estar accediendo continuamente a cada uno de los equipos para realizar instalaciones o pruebas, se comprobó que tener unos nombres tan largos para los nodos era altamente ineficiente. Por tanto se llegó a la solución de denominar a cada uno de los nodos de la subred con la inicial del nombre del cluster, luego a partir de este momento los nodos van del k1 al k16.

La subred saitama como se muestra en la Figura 11cuenta con dos tarjetas de red. La primera de ellas es la que va a conectar a los nodos de la subred con el servidor de la misma forma que están conectados los equipos del subcluster contrario. La única diferencia que existe entre estos y los de kasukabe es que las tarjetas de red de los nodos de saitama son todas de velocidad de 100 Mbps.

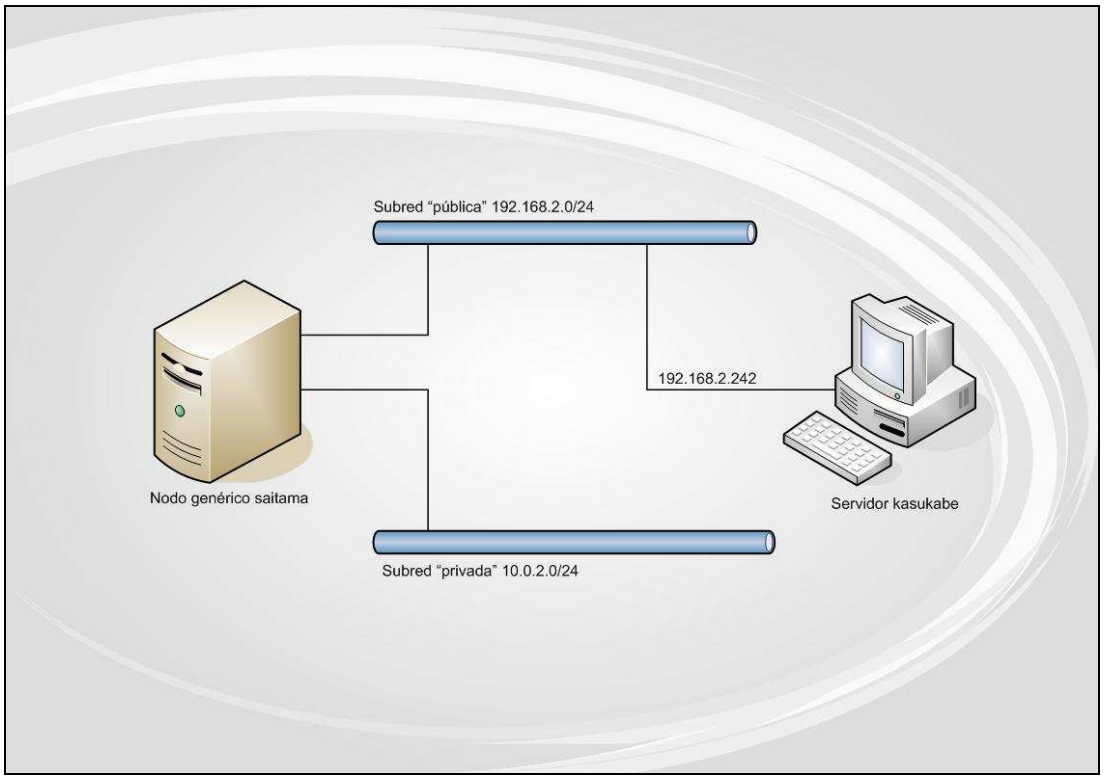

Figura 11: Subred saitama

La segunda tarjeta de red de esta subred, también de velocidad de 100 Mbps, va a interconectar a todos los nodos entre ellos y así se consigue que tengan más de una forma de acceder unos a otros con el fin de probar la eficiencia de esta arquitectura con respecto a la del subcluster kasukabe. El rango de direcciones para estas tarjetas de red va a estar comprendido entre las direcciones 10.0.2.101 y 10.0.2.116.

Los nombres de los nodos de esta subred siguen el mismo formato definido para kasukabe y por tanto los equipos están nombrados entre el s1 (saitama-1, con IP 10.0.2.101) y el s16 (saitama-16, con IP 10.0.2.116).

4.1.2 Almacenamiento compartido

Para este proyecto se habían pensado muchas soluciones de almacenamiento compartido, unas por que ya se conocían y habían sido usadas en parte de la estructuración del Laboratorio del Departamento y otras para realizar diferentes pruebas y observar y estudiar los resultados que se obtuviesen con respecto a las otras tecnologías conocidas.

Para empezar, el sistema de almacenamiento compartido más sencillo y fácil de implantar en la arquitectura de este proyecto es el sistema de ficheros por red NFS. Para utilizar esta tecnología es necesario que el servidor disponga de un directorio o un dispositivo que pueda ser compartido en red por otros equipos. Por tanto se ha tomado una de las particiones del disco duro del servidor como sistema de almacenamiento compartido por NFS. Como se explicará mas detalladamente en la instalación del sistema de ficheros, es tan fácil como indicar en un fichero los directorios que pueden ser compartidos por NFS en el servidor, y en los clientes tan solo hay que montar ese directorio.

Inicialmente se ideó que la configuración ideal sería que para las dos subredes de la estructura tan solo se tuviera una partición compartida por NFS, pero una vez implantado la primera subred (kasukabe), fue requerido por el profesorado de la universidad para ser utilizado para realizar prácticas, para las cuales los nodos del cluster necesitaban poseer su propia partición de NFS para guardar determinados datos. Por ello, y para poder seguir haciendo experimentación y pruebas en el cluster de saitama, se creó una segunda partición en el servidor para ser compartida por NFS.

El segundo sistema de almacenamiento compartido que ya era utilizado en el Laboratorio es *ocfs2* de Oracle sobre el protocolo de comunicaciones iSCSI. Mediante este protocolo se pueden compartir todo tipo de dispositivos, y está un nivel superior a NFS en cuanto a tecnología, pero se estudiará si esta ventaja se ve reflejada también en cuanto a rendimiento. Como se ha explicado en el punto 2.3.2.1, sin comunicación host a host, cada uno de los equipos conectados desconoce cuáles son las intenciones del resto, en cambio esta tecnología si que proporciona esta información. En principio, como se ha estudiado en la documentación del Estado de la Cuestión, se considera que esta tecnología va a ser más eficiente cuando se cuenta con una red de velocidad

Gigabit, pero en este caso se van a compartir dos dispositivos del servidor con cada uno de los clusters con los que se cuenta. Así que lo que se pretende será observar el rendimiento que se obtiene tanto en una red a velocidad Gigabit, como en una red a velocidad de 100 Mbps y de esta forma comprobar si la mejoría que se consigue es grande o si realmente no merece la pena el gasto en hardware con respecto al rendimiento obtenido.

Estos dos sistemas de ficheros utilizan dispositivos compartidos por el servidor kasukabe, cuyos discos duros han sido particionados y dispuestos para este propósito como se muestra en la Figura 12.

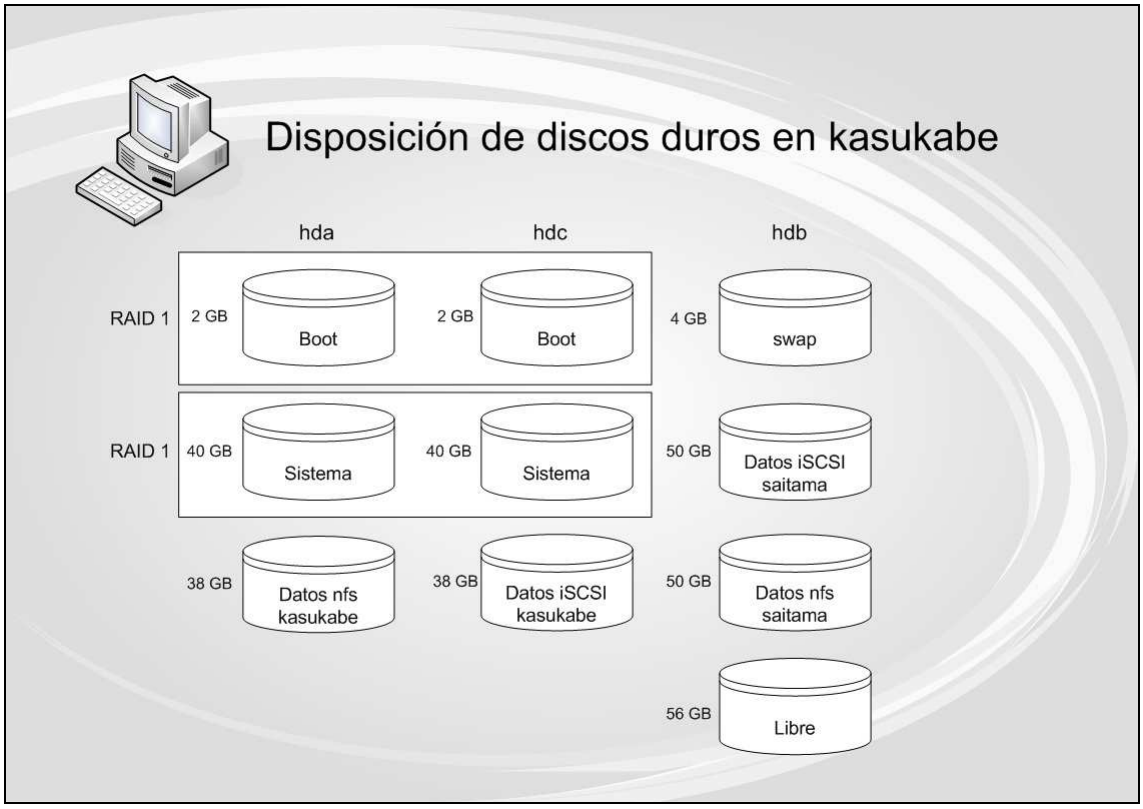

Figura 12: Distribución de discos del servidor

En tercer lugar se desea realizar una comparación de los sistemas de ficheros compartidos anteriores con el Parallel Virtual File System (PVFS). Este sistema no ha sido probado todavía en ningún servidor del Laboratorio del Departamento y será interesante observar y comparar su rendimiento con el que ofrecen en estos momentos tanto NFS como *ocfs2*. Este sistema de ficheros, como se ha explicado en el punto 2.3.4 cuenta con los clientes, con uno o varios servidores de E/S, que serán los que contengan la información en sí y también uno o varios servidores de metadatos, que tienen la

función de indicar a los clientes la localización exacta de la información que desean obtener en cada momento.

Para este proyecto se ha decidido instalar un nuevo disco duro en 4 equipos, realizando una partición en cada uno de ellos orientada a la utilización de este sistema de ficheros. Estos equipos actuarán como de servidores de E/S, haciendo también las veces de servidores de metadatos.

En una implantación común de este sistema de ficheros se suele utilizar un único equipo como servidor de metadatos, pero por razones de seguridad y de alto rendimiento, se ha decidido que este cometido sea llevado a cabo por cuatro equipos para evitar caídas del sistema, por el fallo de un solo nodo, y cuellos de botella.

Por último, se completa el cuarteto de sistemas de ficheros probados en el cluster con Lustre. Al igual que PVFS tampoco ha sido utilizado previamente en el Laboratorio del Departamento, y por ello se debe estudiar muy detalladamente los pasos de compilación e instalación del sistema. Como se ha expuesto en el punto 2.3.3, la estructura del sistema de fichero Lustre es algo compleja, ya que hay muchos actores implicados en su funcionamiento.

En cada uno de los 4 discos duros instalado en los 4 equipos que se indicó para el sistema de ficheros anterior, sobraba una partición. Esta partición es utilizada para actuar como OST (*Object Storage Target*) o almacenamiento de datos, mientras que el primero de los 4 equipos anteriores va a ser considerado como MDS (*Metadata Server*), o servidores de metadatos. El conjunto del cluster, es decir los 32 nodos, harán las funciones de clientes del sistema de ficheros Lustre, y por tanto todos ellos deberán montar la partición ofrecida por los MDS.

4.1.3 Administración y gestión

Para la administración del cluster se han seleccionado varios programas y se han desarrollado diversos scripts. Este sistema de gestión es necesario y altamente recomendable debido a que se han de administrar por separado diferentes aspectos del cluster, como la seguridad, los accesos a las distintas redes, la gestión de archivos, la ejecución de comandos y la gestión de imágenes.

Seguridad

El principal sistema de seguridad implantado el cluster, consiste en la restricción de acceso al cluster como usuario, a no ser que en ese preciso instante se esté ejecutando un trabajo de ese usuario en el equipo en el que desea acceder.

Además se han restringido tanto el tiempo de ejecución como el número de procesos lanzados por los usuarios con el fin de no dejar fuera de servicio al servidor y a los nodos del cluster mediante la modificación del archivo "limits.conf", alojado en el directorio "/etc/security".

Redes e Internet

Se ha configurado el servicio DHCP según el fichero agregado en los Anexos. Se ha implantado de forma que los nodos del cluster siempre tengan una dirección IP fija dentro de la subred de cada uno, asignada directamente por el servidor. Esta aplicación utiliza la dirección física o MAC de cada equipo, asignando a cada una de ellas una dirección IP diferente.

La dirección MAC o *Media Access Control address* (dirección de Control de Acceso al Medio) es un identificador de 48 bits (6 bytes) que corresponde de forma única a una tarjeta o interfaz de red. Es individual y unívoca, y por tanto cada dispositivo tiene su propia dirección MAC determinada y configurada por el Instituto de Ingeniería Eléctrica y Electrónica (IEEE) y por el fabricante. El IEEE asigna a estas tarjetas o interfaces los últimos 24 bits y el fabricante los primeros 24 bits utilizando el OUI (acrónimo del inglés Organizationally Unique Identifier o Identificador Único Organizacional que hace referencia a un número de 24 bits comprado a la Autoridad de Registro del IEEE).

Además se ha desarrollado un script de gestión de red con el que se puede habilitar o deshabilitar la salida de los nodos a Internet. Este script consiste en escribir un cero o un uno en el archivo "ip_forward", alojado en el directorio "proc/sys/net/ipv4", y además lanzar una directiva de las "iptables" con la que habilitar o deshabilitar el enrutamiento de la red.

Gestión de archivos

Con el fin de manejar de forma sencilla la gestión de los archivos dentro del cluster se ha implementado un script con el que se puede copiar uno o varios archivos desde el servidor. Según los parámetros que se le pasen al comando el archivo puede ser copiado a todos los nodos de ambos cluster, a los nodos de un solo subcluster o a un solo equipo de cualquiera de los dos cluster.

Ejecución de comandos

Para poder gestionar la ejecución de cualquier comando remotamente desde el servidor también se ha implementado un script. Inicialmente lo que hace es llamar a otro script que indica si el nodo en el que se quiere ejecutar el comando esta activo y listo, y después de esta comprobación, mediante la aplicación *ssh*, ejecuta el comando solicitado. Al igual que en el caso del script anterior, éste puede ser ejecutado para los dos cluster, para uno de los dos o para un solo equipo de cualquiera de los cluster. Estos scripts han sido agregados en los anexos de este documento.

Colas de trabajos

Uno de los objetivos de este proyecto era poder enviar trabajos a una cola de ejecución para que el cluster trabajase de forma paralela. Para ello se estudiaron varias alternativas diferentes y se decidió tomar una de las dos opciones vistas en los puntos 2.1.4.1, el gestor llamado *Torque*, y 2.1.4.2, el *Condor*. Finalmente se decidió implantar el gestor de colas *Torque* debido a su fácil instalación y su sencilla gestión tanto de los nodos, como de las colas. La instalación de esta herramienta se explica detalladamente en el punto 5.1.2.8.

4.1.4 Virtualización

El sistema de virtualización que se ha utilizado en este proyecto ha sido Xen, por múltiples factores que han apoyado su elección.

En primer lugar, es un sistema conocido y muy utilizado en el entorno del Laboratorio del Departamento de Informática, ya que actualmente todos los servidores que están funcionando para gestionar los servicios, las cuentas de los alumnos, las aulas

informáticas o las páginas Web del departamento, son realmente servidores virtuales creados sobre Xen.

Otro punto fuerte de la elección es la facilidad de uso e instalación de este sistema, ya que a medida que van avanzado las versiones de Debian, se van incorporando más facilidades de instalación y más herramientas de gestión de Xen. Por ejemplo, las primeras instalaciones que se realizaron en el Laboratorio fueron manuales, teniendo que compilar e instalar los paquetes instalables descargados de la página oficial de los desarrolladores del proyecto "http://www.xen.org/" y cambiando diversos archivos de configuración para que el sistema funcionase correctamente. En cambio actualmente basta con seguir el manual que se ha desarrollado en el punto 5.2.1.1.1.

5 DESPLIEGUE DEL CLUSTER

En este capítulo se detallan todos los pasos necesarios para la instalación y configuración de todos los servicios discutidos en el capítulo anterior.

5.1 Servidor

A continuación se especifican las instalaciones que se han llevado a cabo en el servidor, tanto del sistema operativo como de todas las aplicaciones que van a ser necesarias para el correcto funcionamiento del mismo.

5.1.1 Instalación del sistema operativo

Tras la elección de la instalación por red mediante un cd-rom para el arranque se debe comprobar que la BIOS tenga posibilidad de arrancar desde cd-rom e insertar el disco en el lugar correspondiente. Al reiniciar el ordenador, aparece la siguiente pantalla a espera de pulsar una tecla para comenzar la instalación.

Figura 13: Inicio de instalación de Debian Lenny

Como se puede observar en la Figura 13, en las nuevas distribuciones de Linux Debian, se ofrece la posibilidad de hacer una instalación con un pequeño entorno gráfico, y además permite ejecutar una serie de opciones avanzadas que no vienen al caso en este proyecto. En el momento de la instalación de este servidor la versión de Debian que acaba de convertirse en estable es la Lenny, y por tanto es la que se ha utilizado.

Tras pulsar "ENTER" para comenzar el proceso de instalación, se debe introducir el idioma, región y mapa de teclado a utilizar. En nuestro caso, la configuración de este apartado es el habitual para hispanohablantes en España siendo idioma "Spanish", región "España" y mapa de teclado "Español". Posteriormente se realiza la detección del hardware, se analiza el cd-rom y se cargan los módulos necesarios para el correcto funcionamiento del instalador. Seguidamente el instalador detecta la tarjeta de red y adquiere una dirección IP de forma dinámica por DHCP, si no se indica lo contrario.

Tras la obtención de la dirección IP, el siguiente paso es la configuración local del ordenador, para ello se le asigna un nombre que en este caso particular será "kasukabe" y un dominio en caso de tener uno. El dominio del Laboratorio del Departamento de Informática es: "lab.inf.uc3m.es".

A partir de este momento se debe particionar de manera manual los discos para realizar el volcado del sistema base y posteriormente la realizar la instalación del sistema completo. En este caso se dispone de tres discos duros físicos, dos de los cuales disponen de 80 Gb de tamaño y el tercero cuenta con 160 Gb. Por tanto tenemos en total 320 Gb que se han distribuido según se indica en la Figura 14:

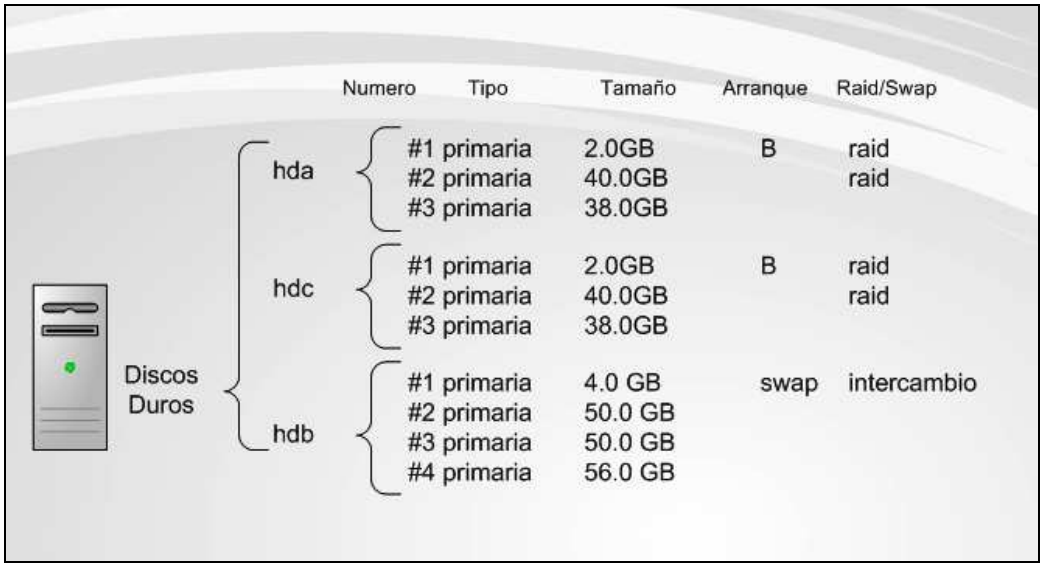

Figura 14: Distribución de particiones del servidor

Entre el primer disco duro y el segundo se han creado dos particiones preparadas para realizar un RAID (originalmente del inglés Redundant Array of Independet Disks,) que se formará entre las 2 primeras particiones de cada disco (RAID 1) y entre las 2 segundas (RAID 1).

En la primera partición de tipo RAID que se crea se almacenará el directorio "/boot" que contendrá la información de arranque del sistema.

La segunda partición de estos dos discos se utilizará para guardar el sistema operativo en si mismo.

Deben escribirse los cambios para que las modificaciones realizadas tengan efecto y posteriormente se crean los multidiscos (MD) para los dispositivos de tipo RAID creados.

Tras este paso, se finaliza el particionado, se configura la zona horaria y se introducen dos datos importantes en el futuro: password de root (contraseña de súper

usuario), usuario y password (Primer usuario del sistema). Tras la configuración de los primeros usuarios se realiza la configuración de la herramienta de instalación de paquetes *apt*, donde se especifica que obtenga la réplica para la instalación del servicio *ftp* de la página de Debian.org en su versión española. Por último se seleccionan los programas y servicios a instalar desde la pantalla de la herramienta *tasksel*, indicando que se instalen básicamente el sistema estándar y el entorno de escritorio.

Después de copiar y configurar todo el sistema base en el sitio especificado se procede a instalar el *grub* que en este caso se realizará en la partición de 2Gb reservada para tal efecto en los primeros discos duros.

En ese momento se concluye la instalación, el cd-rom será expulsado y se reiniciará el ordenador, con lo que el sistema arranca con un sistema operativo Debian instalado correctamente.

5.1.1.1 Configuración básica del servidor

Tras años de instalaciones de sistemas operativos Linux se han establecido una serie de componentes esenciales y necesarios para las tareas de administración de los ordenadores del Laboratorio de Informática.

A continuación se detallan cada uno de los componentes y la configuración típica de cada uno de ellos. En primer lugar, ha de eliminarse (comentarse en este caso) en el fichero de configuración "/etc/apt/sources.list" las entradas relativas a la actualización de paquetes desde el cd-rom. Estas líneas son las siguientes:

```
# deb cdrom:[Debian GNU/Linux testing _Lenny_ - Official Snapshot
i386 NETINST Binary-1 20090127-03:49]/ lenny main
#deb cdrom:[Debian GNU/Linux testing _Lenny_ - Official Snapshot
i386 NETINST Binary-1 20090127-03:49]/ lenny main
```

Este cambio evita problemas a la hora de realizar futuras actualizaciones de paquetes. Como norma general, todos los sistemas deben ser actualizados regularmente y es el primer paso que debe hacerse una vez instalado cualquier ordenador:

```
> apt-get update
> apt-get upgrade
```

A continuación la primera actualización, se procede a instalar todos los elementos básicos obligatorios, para los servidores departamentales, mostrando las configuraciones de cada uno en caso necesario. En primer lugar, hay que instalar y actualizar las configuraciones locales para elegir entre varios idiomas que afectan al juego de caracteres, ordenación alfanumérica, etc. Para llevar a cabo esta instalación deben ejecutarse los siguientes comandos:

```
> apt-get install locales
> dpkg-reconfigure locales
```

Tras el último comando se muestra un menú para las sucesivas elecciones de los locales que han de generarse. Deben ser elegidos todos los que se refieren al idioma español de España (es_ES).

Se elige en la siguiente pantalla de selección la última de las opciones que se han indicado anteriormente (es_ES@euro ISO-8859-15, que son los mapas de caracteres de teclados en español con tecla de Euro).

Después se procede a instalar los paquetes indicados en la Tabla 1:

Nombre	Descripción
less	Lector de ficheros
make	Herramienta de generación y automatización de código.
patch	Herramienta de parcheo de ficheros.
bzip2	Herramienta de compresión de archivos.
xfsprogs	Conjunto de herramientas para el uso de sistemas de ficheros XFS.
vim	Editor de textos.
file	Usado para obtener la descripción y características de ficheros
ftp	Cliente de FTP en consola
lynx	Navegador web en consola
strace	Herramienta de depuración y seguimiento de comandos
nmap	Herramienta de rastreo de puertos TCP y UDP
tcpstat	Herramientas para realizar estadísticas de red
gcc	Compilador básico de C en su versión 4.3

Tabla 1: Aplicaciones básicas del servidor

Todas las herramientas mostradas en la tabla anterior son instaladas de manera directa con el comando:

```
> apt-get install PROGRAMA-A-INSTALAR
```

Tras la instalación de todas estas aplicaciones hay que configurar con especial cuidado el fichero "/etc/network/interfaces", para utilizar las tres tarjetas de red, de la siguiente forma:

```
auto lo
iface lo inet loopback

# Interfaz publica
auto eth0
iface eth0 inet static
        address 163.117.142.242
        netmask 255.255.255.0
        gateway 163.117.142.2

# Interfaz para cluster kasukabe
auto eth1
iface eth1 inet static
        address 192.168.1.242
        netmask 255.255.255.0

# Interfaz para cluster saitama
auto eth2
iface eth2 inet static
        address 192.168.2.242
        netmask 255.255.255.0
```

Teniendo el fichero de esta forma se tiene definida la primera interfaz de red "eth0" con una IP pública y será la que tenga salida a Internet. La segunda interfaz de red tan solo se diferencia de la tercera en el tercer dígito de la dirección IP, ya que ambas interfaces de red son las que conectarán al servidor con cada uno de los dos cluster que se han creado.

Una vez terminada la modificación del fichero anterior hay que reiniciar los interfaces para que utilicen las nuevas direcciones. Este reinicio puede realizarse mediante el comando:

```
> /etc/init.d/networking restart
```

Por último, instalar la herramienta *ssh* (Secure Shell) para acceder a máquinas remotas a través de la red. Este programa se instala de manera análoga a los anteriores:

```
> apt-get install ssh
```

Para conseguir que los ordenadores del cluster tengan plena conectividad entre ellos y no se pida una contraseña cada vez que se quiera cambiar de uno a otro, se debe generar el archivo "authorized_keys" en el directorio de configuración de *ssh*. Para ello se crean primero las claves de tipo RSA y DSA mediante el comando:

```
> ssh-keygen –t rsa
> ssh-keygen –t dsa
```

Se generan de esta forma las claves tanto públicas como privadas, de tipo RSA y DSA y se procede a crear el archivo mediante la siguiente directiva:

```
> cat *.pub > authorized_keys
```

Así se le dice que el contenido de todos los archivos que tengan la extensión ".pub" se guarden en un mismo archivo, el "authorized_keys". Una vez que se tiene esto, basta con copiar el directorio "/root/.ssh" en todos los equipos que se desea que se conecten directamente sin contraseñas.

5.1.2 Configuración avanzada del servidor

Una vez que se tiene instalado en el servidor el sistema base con las aplicaciones y configuraciones básicas para su correcto funcionamiento dentro del dominio del Laboratorio del Departamento de Informática, se procede a instalar todos los programas que van a ser necesarios para el correcto desarrollo del proyecto.

5.1.2.1 iSCSI target

Dentro de Linux hay unas bastantes soluciones software para montar un servidor o *target* iSCSI. La más recomendada es Linux iSCSI Enterprise Target ("http://iscsitarget.sourceforge.net/"). La principal razón es que, hasta la fecha, es la versión más potente, con activo desarrollo y tiene como ventaja que no hay que recompilar el kernel para poder usarla.

Primeramente se van a instalar los paquetes básicos necesarios para que el servidor de iSCSI funcione correctamente:

```
> apt-get install linux-headers-2.6.26-1-686
> apt-get install linux-source libssl-dev
```

Estos paquetes nos permiten indicarle al sistema de almacenamiento iSCSI donde están las cabeceras de Linux y conseguir su pleno funcionamiento.

Se debe descargar la última versión del iSCSI target desde el servidor *sourceforge* (“http://iscsitarget.sourceforge.net/”) que en este caso es la versión “0.4.17”. Se guarda en un lugar seguro, en este caso “/root/apps”, y se descomprime ya que la descarga se produce en formato “tar.gz”.

Una vez realizado el paso anterior se accede a la carpeta que se genera y se abre el archivo “Makefile”, donde debemos indicar a la aplicación dónde encontrará las cabeceras que antes se han instalado:

```
export KSRC := /usr/src/linux-headers-2.6.26-1-686
```

Para continuar con la instalación basta con ejecutar los siguientes comandos desde el directorio base del archivo descomprimido anteriormente:

```
> make
> make install
```

Una vez realizado este paso ya se tendría instalado el servidor de almacenamiento compartido iSCSI básico, y una vez terminada la última línea de la instalación se procede a configurar las propiedades del dispositivo que va a ser compartido en red, modificando el fichero “ietd.conf” alojado en el directorio “/etc”, dejándolo como se observa en los Anexos adjuntados a este proyecto.

En ese archivo se indica el identificador del dispositivo, que debe cumplir siempre el siguiente formato:

```
Target iqn.[yyyy-mm].[nombre del dominio invertido]:[identificador]
```

En la siguiente línea se define el dispositivo que se ofrece como *target*, en la cual se indica la ruta donde está físicamente en el servidor y el tipo de sistema de ficheros. Como se puede ver el dispositivo que va a ser compartido por medio del iSCSI es una

de las particiones de 40Gb que se han dejado indicadas en el apartado 5.1.1 Instalación del sistema operativo.

Después de este paso, se muestra el alias que va a tener el dispositivo en la red y por último, la opción de "InitialR2T" que indica si los iniciadores (clientes del servidor) tendrán que esperar a enviar sus datos hasta que el *target* se los solicite o no. En este caso se desactiva esta opción ya que se pretende que todo funcione en tiempo real con las mejores prestaciones posibles.

Además de este fichero se han de modificar los archivos que ofrecen los permisos para que un cliente del servidor pueda ejecutar un iniciador del iSCSI y por tanto importar la información que este dispositivo contenga. Estos archivos son el "initiator.allow", en el que se especificarán exactamente el rango de equipos que pueden acceder al dispositivo compartido, y el "initiator.deny", en el que se indican los equipos que, a excepción de los que aparecen en el fichero anterior, serán rechazados cuando intenten iniciar el dispositivo.

En este caso el fichero "/etc/initiator.allow" contiene la siguiente línea:

```
ALL 192.168.0.0/16
```

Como ya se ha comentado en los puntos anteriores, la subred que va a pertenecer al subcluster kasukabe comenzará con el rango de IP 192.168.1.X, mientras que el subcluster saitama tendrá como rango de IP el 192.168.2.X. Es por ello por lo que se debe permitir a ambos cluster obtener el *target* con esta línea,

Mientras tanto, el fichero "/etc/initiator.deny" hace que se rechacen el resto de accesos desde Internet:

```
ALL ALL
```

Una vez realizados estos ajustes en la configuración del servidor se puede iniciar el *target* del iSCSI con el siguiente comando:

```
> /etc/init.d/iscsi-target start
```

5.1.2.2 NFS Server

A continuación se explica como ha sido instalado en el servidor el sistema de ficheros en red (NFS) y como se ha configurado para poder ofrecer un dispositivo en red.

En primer lugar, se ha instalado la aplicación como paquete de *apt*, ejecutando el siguiente comando:

```
> apt-get install nfs-kernel-server
```

Una vez realizado este paso necesitamos crear una carpeta para poder montar el dispositivo que se ha dejado reservado para ello en el punto 5.1.1 Instalación del sistema operativo. Para ello se ejecuta el siguiente comando:

```
> mkdir /datos_nfs
```

Después se procede a montar el dispositivo en el directorio recién creado de la siguiente forma:

```
> mount /dev/hda3 /datos_nfs
```

Y para que este montaje se produzca cada vez que se inicia el sistema operativo se inserta la siguiente línea en el fichero "/etc/fstab":

```
# <file system> <mount point>  <type>  <options>    <dump>  <pass>
  /dev/hda3     /datos_nfs      ext3    defaults       0       0
```

Por último, se debe especificar en el fichero "/etc/exports" el dispositivo que se desea compartir por NFS y quién puede importarlo.

```
/datos_nfs 192.168.0.0/16(rw,no_root_squash,no_subtree_check)
```

Para finalizar, se debe reiniciar la aplicación para que arranque con las nuevas configuraciones mediante el siguiente comando:

```
> /etc/init.d/nfs-kernel-server restart
```

5.1.2.3 Cliente NIS

A continuación se procede a instalar la aplicación de *Network Information Service* (NIS). Como se ha explicado en el punto 2.1.3 del Estado de la cuestión, esta aplicación es necesaria para conseguir que los usuarios que tienen cuenta en las aulas de Linux del Laboratorio del Departamento de Informática puedan acceder al servidor mediante alguna aplicación del tipo *ssh* y que conserven en él los datos de su cuenta.

Para empezar se debe instalar el paquete NIS desde la aplicación *apt*, mediante el siguiente comando:

```
> apt-get install nis
```

Al ejecutar este comando, comienza la descarga y posteriormente se procede a la configuración del paquete. Para terminar la instalación aparece una pantalla en la que se debe indicar el dominio al que va a tener acceso el cliente de *nis*. Se debe escribir el dominio del laboratorio: "lab.inf.uc3m.es", y una vez introducido este dato la configuración inicial de la herramienta acaba. Aún así hay que seguir retocando algunos ficheros, como el que se encuentra en "/etc/yp.conf", donde hay que indicarle cuál es el servidor de *nis* del dominio del Laboratorio mediante la siguiente línea:

```
ypserv 163.117.142.230
```

En este caso, la IP que se indica se corresponde con el servidor de "cuentas" desde el cual se ofrece a todos los alumnos el acceso a sus datos personales siempre que tengan cuenta en las aulas de Linux del Laboratorio como ya se ha dicho anteriormente.

También se han de modificar los siguientes ficheros, agregándoles las siguientes líneas:

En "/etc/passwd":

```
+::::::
```

En "/etc/shadow":

```
+::::::::
```

En "/etc/group":

```
+:::
```

Es importante observar como cada una de las líneas son diferentes, ya que cada uno de los puntos de cada archivo se corresponde con una de las características que se pueden definir en ellos pero que se dejan en blanco.

Finalmente para que los usuarios tengan acceso a los datos de las cuentas tienen que importar por NFS el directorio de "/users" del servidor de cuentas. Para ello hay que crear la carpeta mediante el comando:

```
> mkdir /users
```

Después de crear este directorio se agrega la línea siguiente al fichero "/etc/fstab", para que importe por NFS el directorio siempre al arrancar el sistema operativo del servidor:

```
# <file system>   <mount point>    <type>  <options>
cuentas:/users    /users           nfs     rsize=8192,wsize=8192
```

5.1.2.4 SSH

El servicio *ssh* es muy sencillo de instalar ya que se encuentra como paquete en el repositorio de *apt* y es tan fácil como ejecutar el siguiente comando:

```
> apt-get install ssh openssh-server
```

Con esta aplicación instalada ya se puede tanto acceder a otros equipos que permitan el acceso por *ssh*, como acceder a este equipo desde otros.

En caso de querer restringir el acceso a este equipo desde otros, se debe especificar en el archivo "/etc/hosts.allow" el equipo o el rango de IPs que pueden ser aceptados y en el "/etc/hosts.deny", los que serán rechazados. En principio no se considera necesario, y por tanto el acceso al servidor kasukabe será libre desde cualquier equipo para que los usuarios puedan trabajar de forma remota incluso desde su propia casa.

5.1.2.5 Linux Cluster Manager

Este sistema que se encargará de facilitar la gestión del cluster y sus equipos. Antes de comenzar, hay que instalar y configurar una serie de aplicaciones que son necesarias para el correcto funcionamiento del programa.

Para la instalación de las aplicaciones más complicadas se han incluidos los dos puntos seguidos a este en los cuales se explica su instalación y configuración, y a continuación se especifican los de las aplicaciones más sencillas y que no requieren de configuración:

La aplicación *iwidgets* es un servicio que permite el rápido desarrollo de *widgets* para realizar marketing, adaptándolos a los diferentes formatos que aceptan los sitios en los que se pueden instalar. Los *widgets* son pequeñas aplicaciones cuyo objetivo principal es el de facilitar el acceso a funciones usadas frecuentemente y proveer información visual. En nuestro caso será necesario para que se vean correctamente las diferentes opciones de la aplicación LCM.

Su instalación no es demasiado complicada, ya que como muchas otras aplicaciones cuenta con su paquete en el repositorio de *apt* y por tanto basta con ejecutar el siguiente comando:

```
> apt-get install iwidgets4
```

Esta ejecución provoca que se instalen también en cascada las aplicaciones “itcl3” e “itk3”, y estos desencadenan la instalación “tcl8.3” y “tk8.4”. Automáticamente al instalarse quedan configuradas y listas para usarse.

Una vez realizado el paso anterior y después de instalar y configurar correctamente los servicios de DHCP y TFTP (puntos 5.1.2.5.1 y 5.1.2.5.2) se procede a instalar la aplicación LCM en si. Como no dispone de paquete desde el repositorio de *apt* se debe descargar de la página oficial el paquete instalable que se encuentra en la dirección “http://linuxcm.sourceforge.net”.

Se selecciona la página correspondiente a descargas y ahí se encuentran las últimas versiones tanto de la documentación como de los archivos instalables del programa. Dentro de los instalables se pueden encontrar tanto el archivo

correspondiente al servidor como el de los clientes. Se procede a descargar el indicado para el servidor que tiene el nombre de **lcm-source-2.90_2.i386.deb** y se guarda en el equipo, en la carpeta "/opt".

Para instalar la aplicación se debe ejecutar el siguiente comando:

```
> dpkg -i lcm-source-2.90_2.i386.deb
```

Después de realizar este paso la aplicación queda instalada y configurada correctamente en el servidor.

Al ejecutar el comando *lcm* en este equipo aparece la siguiente pantalla inicial:

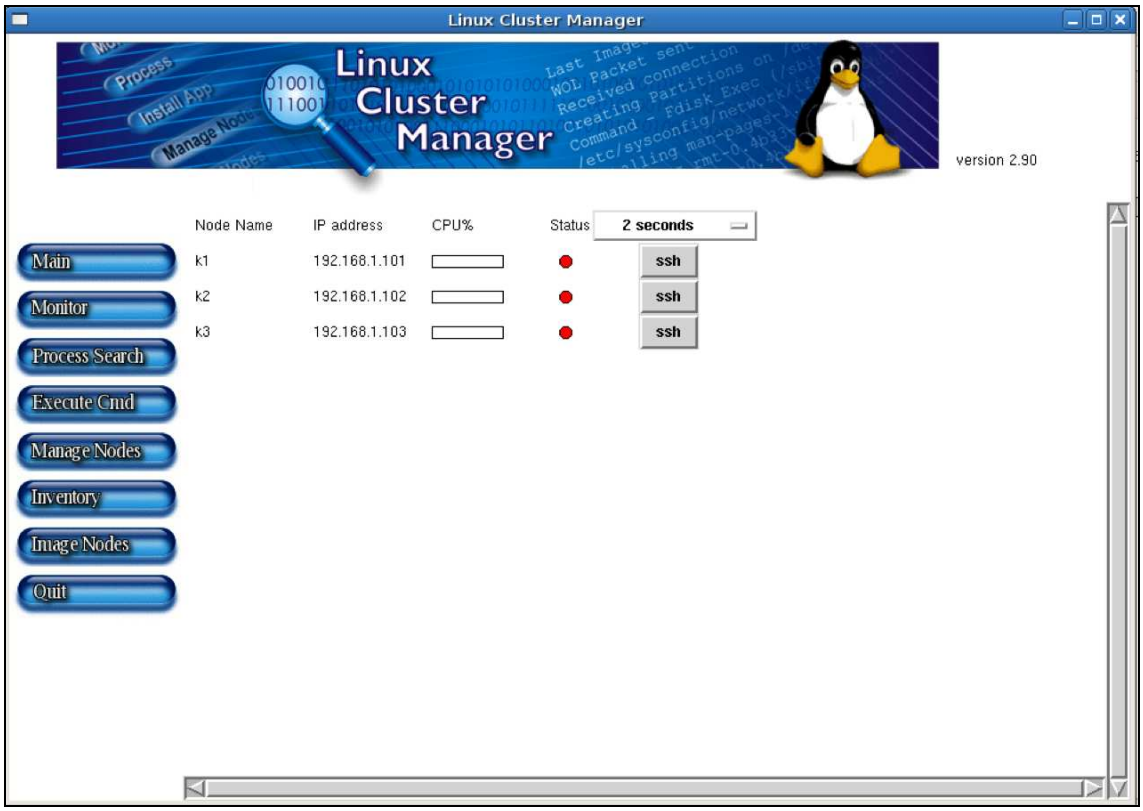

Figura 15: Pantalla inicial de LCM

5.1.2.5.1 Servicio DHCP

El servicio DHCP, como ya se explicó anteriormente, es un protocolo de red que permite a los nodos de una red obtener sus parámetros de configuración automáticamente, como por ejemplo la IP.

Para su instalación tan solo es necesaria la ejecución del siguiente comando:

```
> apt-get install dhcp3-server
```

Una vez terminado de instalar se procede a configurarlo con las características mas apropiadas para este caso. Lo primero que se debe hacer es cambiar el archivo de configuración, ubicado en "/etc/default/dhcp3-server", modificando la siguiente línea:

```
INTERFACES="eth1 eth2"
```

Con este cambio se consigue que el servidor de direcciones escuche por las interfaces uno y dos, que son las que están conectadas a los sub-cluster de kasukabe (eth1) y de saitama (eth2). Así cada uno de los equipos de estos dos cluster pedirán sus direcciones de red al servidor, y éste se las asignará directamente utilizando el archivo de configuración localizado en "/etc/dhcp3/dhcpd.conf", agregado a los Anexos del presente documento.

5.1.2.5.2 Servicio TFTP

Como ya se comento anteriormente TFTP es un protocolo de transferencia muy simple semejante a una versión básica de *ftp*. A menudo se utiliza para transferir pequeños archivos entre ordenadores en una red. Para el correcto funcionamiento del servidor se debe instalar como debe instalarse como paquete, ya que está en el repositorio de paquetes de la aplicación *apt*. Su instalación es sencilla, ya que tan solo se debe ejecutar la siguiente sentencia:

```
> apt-get install tftpd-hpa
```

Una vez realizada esta instalación se procede a configurar el programa, principalmente modificando las siguientes líneas en el fichero de configuración "/etc/defaults/tftpd-hpa":

```
Run_daemon="yes"
Options='-l -s /tftpboot'
```

Por tanto lo que se le indica es que el demonio que ejecuta este programa de forma continua en el servidor este activado, y además se le especifica el lugar en el cual se van a encontrar los archivos que se van a compartir.

5.1.2.6 OpenMP

La librería OpenMP, que como ya se explicó anteriormente, sirve para añadir concurrencia a las aplicaciones mediante paralelismo con memoria compartida, se instala directamente al instalar la última versión del compilador general de C en Linux, la librería *gcc* en su versión 4.3.

Para que esta aplicación funcione correctamente en el entorno del servidor, se realiza un enlace simbólico como se muestra a continuación:

```
> ln -s /usr/bin/gcc-4.3 /usr/bin/gcc
```

5.1.2.7 MPICH2

La aplicación MPICH2, como ya se explicó en el capítulo Estado de la Cuestión, se va a instalar en su versión mas reciente hasta la fecha actual. La versión 1.0.8 es la elegida ya que la versión posterior aún no es la estable y no se desea correr riesgos innecesarios.

La instalación es sencilla, pero es recomendable instalar el código fuente disponible en la Web oficial del grupo de desarrollo de MPICH:

```
http://www.mcs.anl.gov/research/projects/mpich2/downloads/index.php
?s=downloads
```

Una vez descargado el paquete que contiene todas las fuentes (se ha optado por el directorio "/opt"), se procede a descomprimirlo debido a que se encuentra comprimido en un archivo con la extensión "tar.gz".

```
> tar -zxvf mpich2-1.0.8.tar.gz
```

La ejecución del comando anterior creará una nueva carpeta con el mismo nombre que el archivo descargado, pero omitiendo la extensión. Dentro de esta carpeta se encuentran tanto el archivo de configuración del sistema instalable, "configure", como los archivos de instalación propiamente dichos. Se comienza con la configuración, necesaria para crear el resto de archivos de instalación:

```
> ./configure --with-mpe --enable-g=all
```

El primer argumento indica que deseamos generar MPICH2 con el paquete extra de trazado y visualización. El segundo argumento indica que prepararemos MPICH2 para que sea compatible con todas las opciones de depuración.

Después de tener configurado el compilador para las características propias del equipo, se procede a ejecutar los archivos que compilarán e instalarán el programa correctamente en el sistema:

```
> make
> make install
```

5.1.2.8 Torque

Como ya se ha explicado en el punto 2.1.4.1, *Torque* es una herramienta que no está disponible como paquete y por tanto, no puede ser descargada mediante la aplicación *apt*. Esta descarga se debe realizar desde la página oficial de la empresa que desarrolla *Torque*: "www.clusterresources.com/downloads/torque".

Una vez descargada la última versión se almacena en el directorio del servidor "/opt", donde se guardan normalmente las aplicaciones que no se instalan como paquetes en los servidores. Se debe descomprimir el archivo ya que al descargarlo está en formato "tar.gz", mediante el comando:

```
> tar -zxvf torque-2.4.0b1.tar.gz
```

Con la ejecución del comando anterior se obtiene una nueva carpeta con el nombre "torque-2.4.0b1", en la cual podemos encontrar tanto el archivo de configuración "configure", como los archivos de compilación de la instalación.

Para comenzar con la instalación de Torque se deben instalar previamente una serie de aplicaciones que son absolutamente necesarias para el correcto funcionamiento de la herramienta. Éstas son:

```
> apt-get install libpam0g-dev g++
```

Una vez realizada esta tarea se procede a ejecutar el archivo de configuración mediante el siguiente comando:

```
> ./configure --enable-server --enable-clients --with-scp
-with-default-server=kasukabe --without-tcl --with-pam
--prefix=/opt/torque
```

De los argumentos necesarios para ejecutar "./configure" quizás el más importante sea "--with-pam". Este argumento indica que deseamos generar Torque con soporte para *Pam*. El modulo *Pam* será usado para impedir el acceso directo de los usuarios a los nodos del cluster.

La ejecución de este archivo de configuración realiza una serie de comprobaciones en el sistema operativo del servidor para asegurarse de que la instalación no tendrá ningún problema y que la herramienta funcionará correctamente.

Una vez realizada esta comprobación se comienza la generación de verdad, ejecutando los siguientes comandos desde la carpeta antes indicada:

```
> make
> make install
```

Al terminar este paso el servidor de Torque quedará instalado y listo para funcionar en el equipo.

5.2 Cluster

A continuación se explican las instalaciones tanto del sistema operativo, como de las aplicaciones más importantes, además de las diferentes configuraciones para cada una de ellas. El objetivo en este punto es conseguir un equipo con las menores

instalaciones posibles para que funcione de forma ligera, pero a su vez que sea completo.

5.2.1 Instalación básica del cliente

El primer paso que se debe realizar con un nodo del cluster es instalar el sistema operativo, aplicándole las características que se consideren necesarias, y una vez se tiene el equipo funcionando correctamente se procede a instalarle las aplicaciones que se van a necesitar para el proyecto.

El sistema operativo con el que cuentan cada uno de los nodos es una distribución de Debian Lenny, ya que como se explicó en el punto 5.1.1, es la versión que acaba de convertirse recientemente en la distribución estable. Este punto no requiere demasiadas explicaciones ya que esta instalación se ha realizado tan solo con los paquetes más básicos.

El apartado más destacable de la instalación de los equipos que forman parte del cluster es la distribución del disco duro que se ha llevado a cabo. Estos equipos cuentan con unos discos duros de 40 Gb, y se han particionado de forma que han quedado repartidos según la Figura 16:

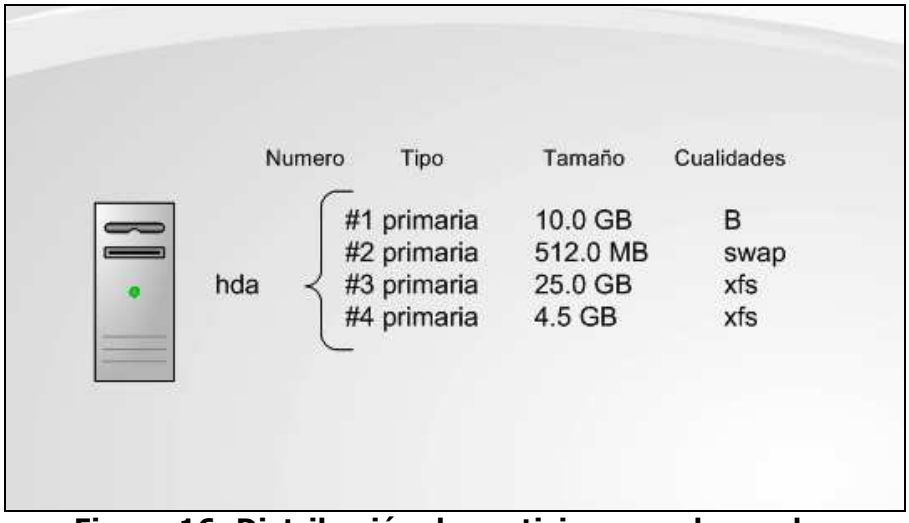

Figura 16: Distribución de particiones en los nodos

En la primera partición de 10 GB es donde se encuentra tanto el sistema de arranque como el sistema operativo propiamente dicho. Se ha considerado más que

suficiente contar con este tamaño, aunque una vez instalado el sistema se ha observado como con mucho menos espacio hubiera bastado.

En la siguiente partición se han dispuesto 512 MB de espacio de intercambio, o *swap*, que se ha considerado suficiente para la memoria de la que disponen los equipos.

La tercera partición es un espacio que se ha reservado para guardar datos que puedan ser necesarios y que se quieran separar de la partición del sistema por seguridad.

Por último los 4,5 GB de almacenamiento restantes se considerarán como partición no utilizada, ya que esa partición va a ser usada por las máquinas virtuales como almacenamiento de datos.

Se han formateado estas dos últimas particiones en el sistema de ficheros XFS, ya que se ha considerado que para almacenamiento de grandes archivos este sistema de ficheros es más rápido que “ext3”, y además así se “desperdicia” menos espacio que este sistema de ficheros, ya que “ext3” tiene que reservar algo de espacio para los árboles de directorios.

A partir de este punto se explica cómo se han realizado las instalaciones de las dos aplicaciones más importantes en los nodos físicos.

5.2.1.1.1 Instalación de Xen

Para realizar la instalación del sistema de paravirtualización Xen, se requiere antes de nada la instalación de varios paquetes básicos. Algunos de ellos son necesarios para el correcto funcionamiento de la herramienta, mientras que otros son instalados para poder realizar controles sobre el rendimiento tanto del equipo, como de las maquinas virtuales creadas en él.

Estos paquetes son los siguientes:

Paquete	Descripción
less	Lector de ficheros
make	Herramienta de generación y automatización de código.
patch	Herramienta de parcheo de ficheros.
bzip2	Herramienta de compresión de archivos.
xfsprogs	Conjunto de herramientas para el uso de sistemas de ficheros XFS.
vim	Editor de textos.
file	Usado para obtener la descripción y características de ficheros
ftp	Cliente de FTP en consola
lynx	Navegador Web en consola
strace	Herramienta de depuración y seguimiento de comandos
nmap	Herramienta de rastreo de puertos TCP y UDP
tcpstat	Herramienta para realizar estadísticas de red
nfs-kernel-server	Herramienta que permite importar y exportar dispositivos en red mediante NFS
ssh	Aplicación de acceso seguro a los nodos por consola
gpm	Herramienta que habilita la utilización del ratón en el modo consola.
gcc	Compilador básico para lenguaje C

Tabla 2: Aplicaciones básicas de los nodos

Todas las herramientas mostradas en la Tabla 2 son instaladas de manera directa con el comando:

```
> apt-get install PROGRAMA-A-INSTALAR
```

Una vez finalizada la instalación de estos paquetes básicos se procede a instalar los paquetes que realmente harán que la aplicación funcione en el equipo.

```
> apt-get install bridge-utils xen-hypervisor-3.2-1-i386
> apt-get install xen-linux-system-2.6.26-1-xen-686 xen-tools
```

Una vez realizado el paso anterior, se modifica el fichero “xend-config.sxp”, situado en el directorio “/etc/xen/” para que quede como se muestra en el anexo adjuntado con este documento.

Por último se debe reiniciar el equipo y arrancar seleccionando el kernel de Xen en el *grub* del sistema operativo, con lo que ya se tendrán disponibles todas las herramientas que Xen proporciona para la creación y gestión de máquinas virtuales.

5.2.1.1.2 Instalación del cliente LCM

La instalación del cliente de la aplicación Linux Cluster Manager es muy sencilla, ya que únicamente hay que descargarse el instalador de la página principal del proyecto: "http://linuxcm.sourceforge.net".

En nuestro caso ese instalador se llama "**lcm-client-source_2.90_10.i386.deb"** y como todas las aplicaciones que no se instalan directamente como paquete, se guarda en la carpeta "/opt".

Antes de su instalación en el equipo es son necesarias varias aplicaciones básicas para su correcto funcionamiento, como son:

- TCL 8.4
- TK 8.4
- iwidgets 4.0.1
- iTCL 3.3

Al incluirse todas estas aplicaciones como dependencias del paquete "*iwidgets4*", basta con instalar éste y los demás quedarán correctamente instalados en la máquina.

La instalación del fichero descargado desde la página oficial de la aplicación se basa tan solo en la ejecución del siguiente comando:

```
> dpkg -i lcm-client-source_2.90_10.i386.deb
```

Para finalizar, se debe puntualizar que esta instalación es muy rápida, ya que dura menos de un segundo, y con ello ya esta listo el sistema operativo para poder ser manejado directamente por la aplicación del servidor.

5.2.2 BIOS

Hay varias cosas que se deben tener presentes antes de configurar la BIOS de los nodos, y lo primero de todo, es que se van a tener que configurar todos a mano. La herramienta de administración del cluster no tiene ninguna opción pueda servir de ayuda para esta tarea, es mas, se debe configurar la BIOS previamente para poder hacer uso de dicha herramienta.

Lo segundo a tener presente es recordar que, aunque se va a disponer de una pantalla y un teclado para poder acceder a los nodos directamente si fuera necesario, la mayor parte del tiempo los nodos van a funcionar sin ningún periférico conectado. Se debe controlar en la BIOS cambiando la opción de “HALT ON” asignándole el valor "NO ERROR", y así el ordenador no se quedara bloqueado al arrancar.

Tras esto, lo único que queda es configurar la tabla de arranque del sistema. Es importante que quede activado el *floppy* (disquetera) como primer elemento de arranque, ya que será necesario para poder instalar las imágenes que se irán copiando vía PXE. Para esta instalación los equipos deberán iniciarse con un disquete de arranque. Más adelante se explicará tanto el proceso de instalación como la creación de dicho disquete de arranque.

Como segundo elemento en la tabla de arranque se activará el disco duro del equipo, el resto no intervienen en el proceso y se pueden dejar deshabilitados.

5.2.3 Creación del disquete de arranque

Dado que los nodos del cluster no disponen de la opción “Wake on Lan”, y como además la BIOS tampoco proporciona la opción de arranque vía PXE, se ha decidido crear unos discos de arranque ya que por las especificaciones de las tarjetas de red se sabía que eran capaces de ello, aunque no se reflejara en la BIOS de los nodos.

Las investigaciones que se llevaron a cabo con respecto de la creación de un disco de arranque PXE se dirigieron directamente a la página “www.etherboot.org”, donde se descubrió que existe un paquete para Debian, denominado “etherboot”, que puede descargarse y utilizarse fácilmente. Una vez instalado en el sistema, se deben ejecutar los siguientes comandos para crear una imagen del disquete de arranque.

```
> make bin/gpxe.dsk
> cp bin/gpxe.dsk gpxe.dsk
```

Una vez hecho esto, solo queda trasladar esta imagen a un disquete en blanco, mediante este comando.

```
> dd if=bin/gpxe.dsk of=/dev/fd0
```

Como se ha indicado anteriormente, las BIOS de los nodos del cluster están configuradas para que el disquete sea el primero en la tabla de arranque. Se introduce el disco de arranque en el nodo destino, y lo apagamos. Al volver a encenderlo se iniciará con el arranque del disco y el LCM en el servidor detectará el arranque vía PXE, dando paso al proceso de instalación. Para concluir, señalar que el disquete de arranque, debe permanecer introducido, durante todo el proceso de instalación.

5.2.4 Distribución de imágenes con LCM

Una vez que se tiene instalada y configurada correctamente la aplicación tanto en el servidor como en el cliente origen, la distribución de esta imagen es muy sencilla. Tan solo hay que ejecutar el comando *lcm* en el servidor y se puede empezar a interactuar con la aplicación. Se recuerda que es muy importante para el correcto funcionamiento de esta herramienta que los servicios DHCP y TFTP estén perfectamente instalados y configurados según los parámetros expuestos en los puntos 5.1.2.5.1 y 5.1.2.5.2 respectivamente.

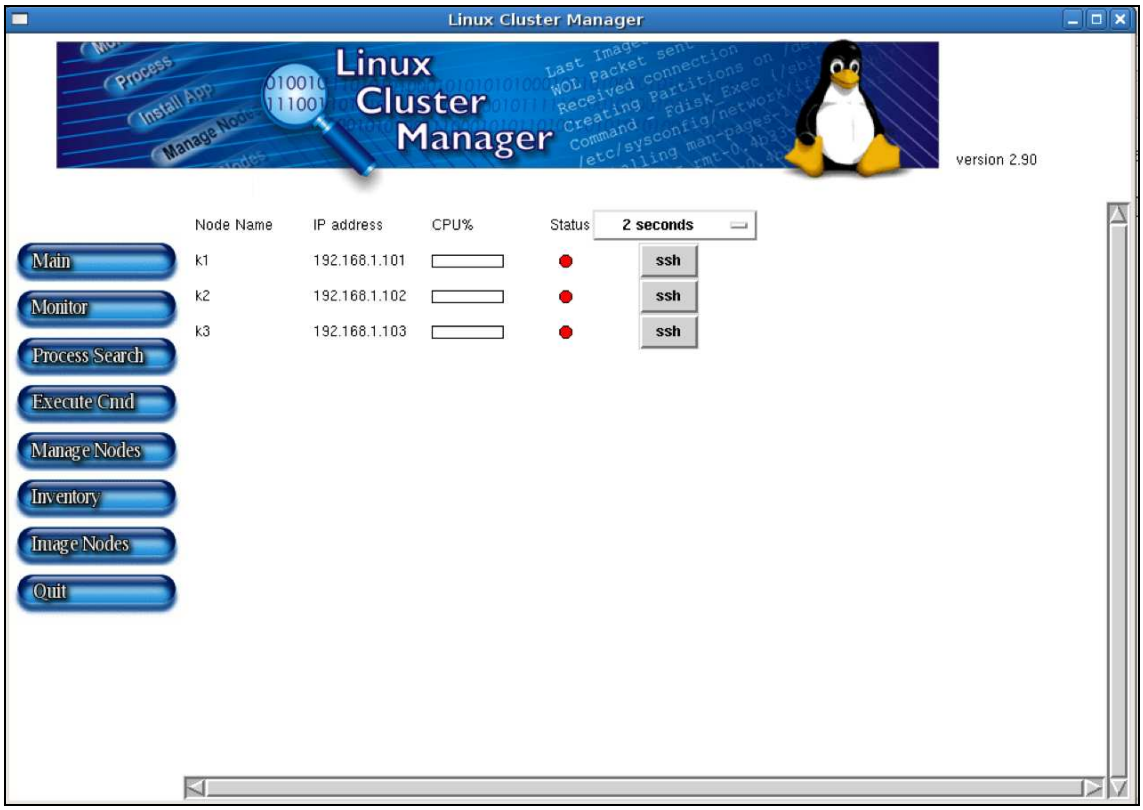

Figura 17: Pantalla inicial del LCM

Hay que seleccionar la opción de "*Image Nodes*" en el menú de la izquierda y a continuación pinchar sobre la pestaña superior que dice "*Create Image*". Aparece la vista del programa según la Figura 18:

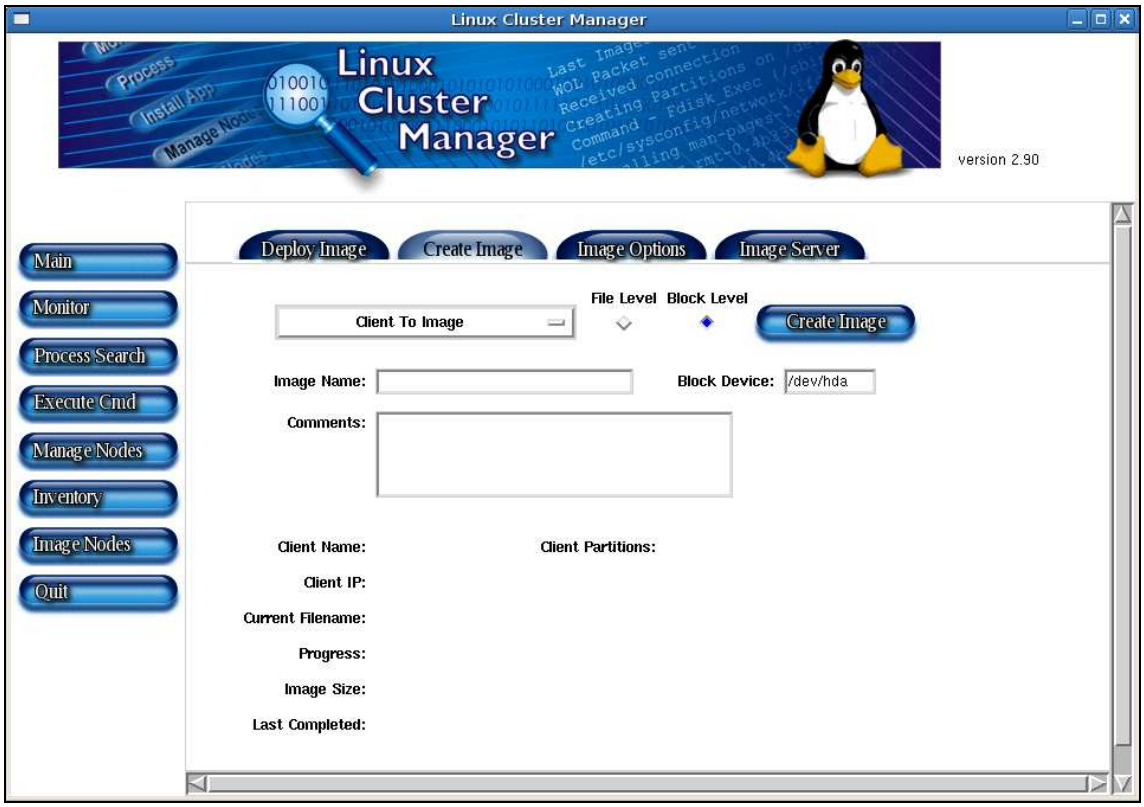

Figura 18: Pantalla de creación de imágenes del LCM

Donde se observa la leyenda "*Client To Image*" se despliega una pestaña donde se pueden seleccionar todos los equipos pertenecientes a la red cuyas imágenes pueden ser capturadas por esta aplicación.

LCM, permite realizar dos tipos diferentes de imágenes, a nivel de fichero o a nivel de bloque. Se ha decidido realizar la captura de imagen por el "*File Level*", por dos razones fundamentales. La primera de ellas es una cuestión de comodidad ya que las imágenes tomadas a nivel de fichero tardan menos tiempo, tanto en su creación como en su instalación, no son tan rígidas como las capturadas a nivel de bloque, ocupan menos espacio en el servidor y requieren menos espacio a la hora de instalarse en el equipo receptor. La segunda razón, y realmente la mas importante es que el "*Block Level*" requiere PXE y "*Wake on Lan*" (arranque en red) para su funcionamiento, tanto a la hora de crear las imágenes como en su instalación. Las imágenes que son recogidas a nivel de fichero pueden crearse mientras el nodo origen esta activo, por lo que no hay que dar de baja el sistema para ello, lo que supone una gran ventaja. En cualquier caso y aunque los dos tipos de imágenes necesitan realizar su instalación vía PXE, los nodos de este cluster tienen un pequeño inconveniente, y es que no soportan la opción de "*Wake*

on Lan", por lo que las imágenes de "*Block Level*" estuvieron descartadas desde el inicio.

La copia de la imagen en el servidor dura aproximadamente cinco minutos de reloj. Según las instalaciones que tenga el nodo origen desde el que se vaya a copiar la imagen, esta imagen tendrá un tamaño variable. En este caso concreto al tener una instalación avanzada el tamaño que ocupa comprimida suele ser de entre 700 y 800 MB. En este caso, al tener creada una máquina virtual en una partición del disco duro (, la imagen en cuestión ocupa prácticamente el doble de lo que ocuparía una imagen en condiciones normales.

Una vez terminado el paso de la imagen del cliente al servidor, se pueden comprobar las características de esta imagen en la pestaña "*Image Options*".

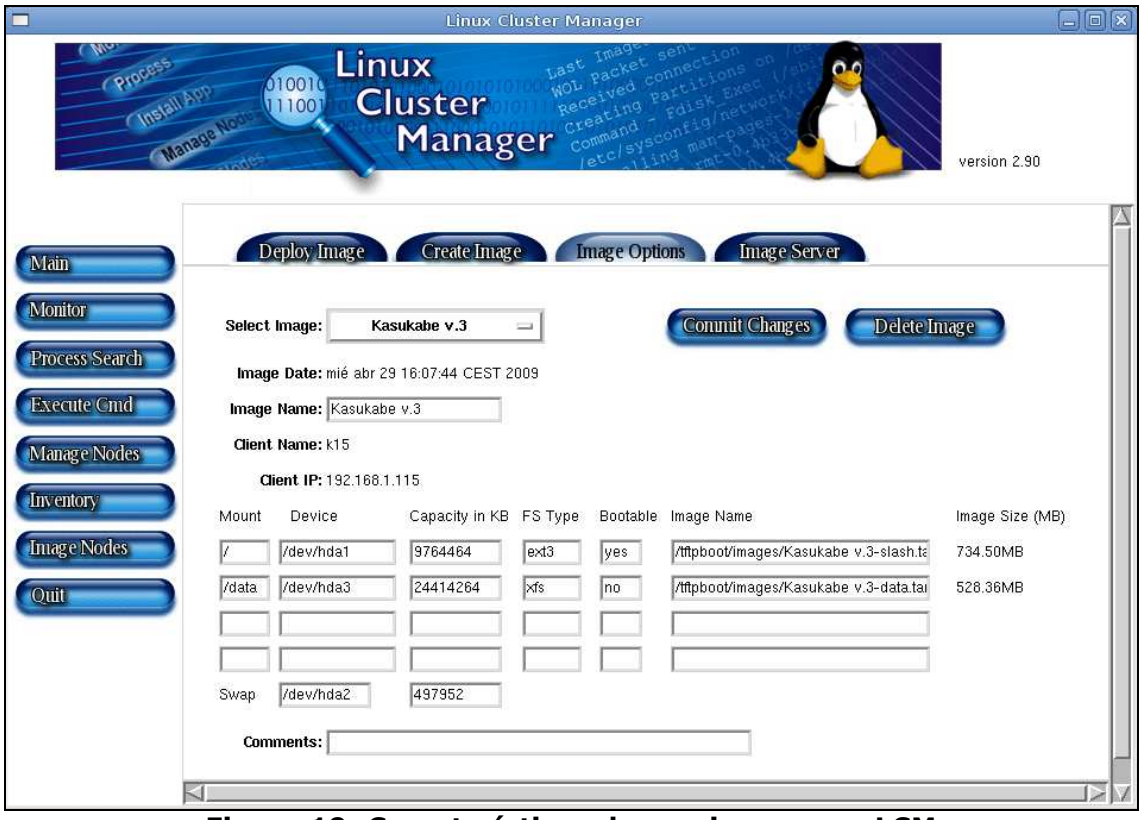

Figura 19: Características de una imagen en LCM

Implantar esta imagen en otro equipo requiere la ejecución de varios pasos. El primero de ellos es añadir el nuevo equipo en el conjunto de los que pueden ser manejados por la aplicación. Para ello basta con ir a la pantalla de administración de nodos, ("*Manage Nodes*") en el menú de la izquierda. Aparece la siguiente pantalla,

donde para crear un nuevo equipo se debe seleccionar con el botón derecho del ratón la palabra "*Hosts*". Aparece un menú donde se selecciona "*Add Host*", y se le cambia el nombre para poner el que realmente va a tener el quipo en el cluster que se está construyendo.

Una vez realizado este paso se procede a pasar la imagen al equipo desde la pestaña "*Deploy Image*", en la ventana de "*Image Nodes*". Se debe tener mucho cuidado ya que al seleccionar este procedimiento, aparece marcada por defecto la instalación mediante "*Wake on Lan*" ("*Issue WOL*"). Se debe deseleccionar, y se busca en la lista de imágenes disponibles que aparece en el desplegable que se encuentra a su lado la que se desea implantar en el equipo. Tras esto, se selecciona el equipo que va a recibir la imagen nueva y se pincha sobre el botón "*Execute*". Esta acción provoca que la aplicación, como servidor de imágenes, se quede esperando el arranque de la máquina seleccionada para comenzar su instalación.

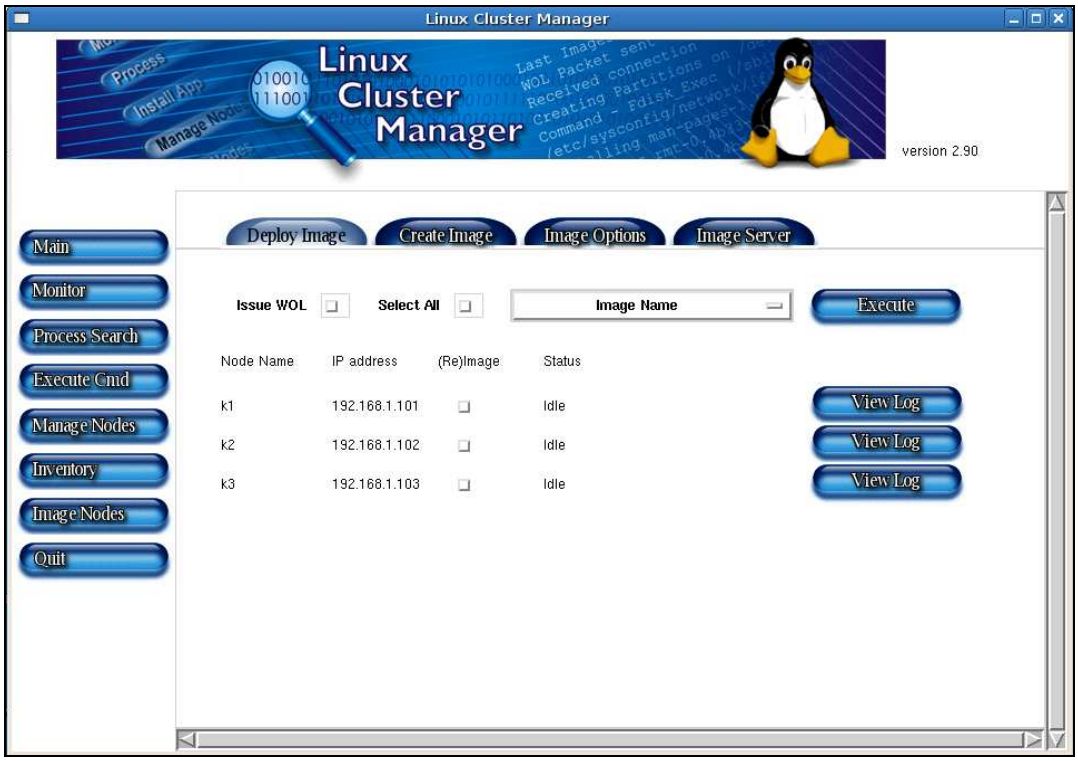

Figura 20: Pantalla de instalación de imagen en LCM

5.2.5 Creación de máquinas virtuales

La creación de una máquina virtual, gracias a la herramienta de virtualización Xen, debe seguir unos pasos determinados. Para este cometido se ha creado un pequeño

script que se puede encontrar en los Anexos añadidos junto con este documento y con el que automáticamente se genera una nueva máquina básica con las características que han sido indicadas. El primer paso que se debe realizar sobre esta máquina es cambiarle la contraseña de súper-usuario (root) ya que inicialmente la máquina ha sido creada sin contraseña, lo que supone un gran agujero de seguridad. Para ello se debe aplicar el siguiente comando:

```
> passwd root
```

Esta ejecución provoca que el sistema pida por teclado la inserción de la nueva contraseña de súper-usuario, en dos ocasiones para asegurar la correcta escritura de la misma.

Después de realizar este paso y se tiene la máquina lista para funcionar se procede a su configuración avanzada.

5.2.6 Configuración avanzada de nodos y máquinas virtuales

A continuación se explican las diferentes configuraciones e instalaciones avanzadas que han sido realizadas tanto en los nodos del cluster como en las máquinas virtuales que estos nodos contienen.

5.2.6.1 Configuración del almacenamiento compartido

5.2.6.1.1 NFS

Una vez se tiene configurado el servidor de NFS en el equipo que va a exportar sus directorios en la red, la instalación del cliente es muy básica. En primer lugar se utiliza la herramienta de descarga de paquetes "*apt*" para instalar los archivos comunes de NFS, ejecutando el siguiente comando:

```
> apt-get install nfs-common
```

Teniendo la aplicación instalada en el equipo basta con cambiar el fichero "fstab" de la máquina, agregando las siguientes líneas:

```
#Datos NFS
# <file system>             <mount point> <type> <options> <dump> <pass>
192.168.2.242:/datos_nfs_saitama /datos_nfs nfs defaults       0       0
```

5.2.6.1.2 Cliente iSCSI

La instalación del sistema de almacenamiento compartido en red *ocfs* sobre iSCSI es ligeramente mas complicada que la llevada a cabo en el punto anterior. Para empezar, como es habitual, se procede a la instalación de la herramienta mediante los paquetes básicos, ejecutando las siguientes directivas:

```
> apt-get install open-iscsi ocfs2-tools
```

Una vez se tiene el sistema de ficheros en el equipo se procede a configurarlo, siguiendo los siguientes pasos.

En primer lugar se debe cambiar el nombre que va a tomar la máquina en el registro del servidor, a la hora de conectarse a él. Ese nombre se encuentra en el fichero "/etc/iscsi/initiatorname.iscsi" y debe seguir un formato predeterminado, empezando por el prefijo de la aplicación "iqn", seguido por el año y el mes en que ha sido añadido el nodo al cluster, y terminando con el nombre del nodo y su dominio correspondiente, pero escrito de forma inversa.

En este ejemplo se puede observar como debería de quedar el nombre de un equipo:

```
InitiatorName=iqn.2009-03.es.uc3m.inf.lab.cluster.mv1
```

A continuación se modifica el fichero de configuración que toma los valores del *ocfs* por defecto. Este fichero se encuentra en la ruta "/etc/default/o2cb" y debe tomar los valores que se indican en los Anexos agregados a este documento.

El siguiente paso es descubrir por parte del cliente las particiones que son compartidas por el servidor mediante el sistema de ficheros iSCSI. Para ello se deben ejecutar los siguientes comandos desde el subcluster de kasukabe:

```
>iscsiadm -m discovery -t sendtargets -p 192.168.1.242:3260
>iscsiadm -m node -T iqn.2009-01.es.uc3m.inf.lab:Datos_cluster_kasukabe
-p 192.168.1.242 -l
>iscsiadm -m node -T iqn.2009-01.es.uc3m.inf.lab:Datos_cluster_kasukabe
-p 192.168.1.242   --op update -n node.startup -v automatic
>iscsiadm -m node -T iqn.2009-01.es.uc3m.inf.lab:Datos_cluster_kasukabe
-p 192.168.1.242   --op update -n node.conn[0].startup -v automatic
```

Y desde el subcluster de saitama se ejecutarían los siguientes:

```
>iscsiadm -m discovery -t sendtargets -p 192.168.2.242:3260
>iscsiadm -m node -T iqn.2009-02.es.uc3m.inf.lab:Datos_cluster_saitama
-p 192.168.2.242 -l
>iscsiadm -m node -T iqn.2009-02.es.uc3m.inf.lab:Datos_cluster_saitama
-p 192.168.2.242   --op update -n node.startup -v automatic
>iscsiadm -m node -T iqn.2009-02.es.uc3m.inf.lab:Datos_cluster_saitama
-p 192.168.2.242   --op update -n node.conn[0].startup -v automatic
```

Una vez se han recuperado los dispositivos compartidos por medio de iSCSI por parte del servidor, y después de haber registrado el nodo en el cluster, se procede a reiniciar la aplicación para que tome los nuevos valores indicados.

```
> /etc/init.d/open-iscsi restart
```

Una de las consecuencias de ejecutar este comando es que la nueva partición agregada al equipo pasa a estar disponible en el directorio de los dispositivos "/dev" con el nombre de dispositivo "sda". El siguiente paso que se debe realizar es el formateo de dicha partición, para que acepte el sistema de ficheros **ocfs2**.

```
> mkfs.ocfs2 --fs-feature-level=max-features –N 16 /dev/sda
```

Se consigue de esta forma tener un sistema de ficheros distribuido, pero para el equipo toma el rol de sistema de almacenamiento local. Para que este dispositivo sea montado siempre que arranque el ordenador, se debe incluir la siguiente línea en el "fstab":

```
#Datos iSCSI
/dev/sda        /datos_cluster_iscsi    ocfs2   _netdev 0       0
```

Por último se deben incluir en el archivo "cluster.conf", alojado en el directorio "/etc/ocfs2", todas las entradas de los nodos que van a formar parte del cluster.

5.2.6.1.3 PVFS

La instalación del sistema de ficheros distribuido PVFS consta de varios pasos que se deben llevar a cabo para conseguir que el sistema quede instalado en el equipo como un módulo del kernel. Cabe destacar que la instalación de este sistema de ficheros ha sido realmente costosa, ya que inicialmente se deseaba realizar sobre el kernel de las máquinas virtuales, que está preparado para ser virtualizable con Xen. Se estudiaron múltiples soluciones a este problema, incluso parcheando el kernel del sistema operativo para que cogiese el módulo, pero finalmente no se consiguieron resultados satisfactorios, por lo que esta instalación se ha desarrollado íntegramente sobre el kernel de Linux original del equipo.

El primer paso que se debe realizar es buscar los paquetes que PVFS necesita para funcionar correctamente. Las primeras dependencias que se han encontrado revisando la documentación ofrecida en la página oficial del sistema de ficheros "www.pvfs.org" han sido las librerías de bases de datos desarrolladas por la Universidad de Berkley. Para su instalación se deben aplicar las siguientes directivas:

```
> apt-get install db4.6-util libdb4.6 libdb4.6-dev
```

Además, para poder instalar este sistema de ficheros y aplicarlo como módulo en el sistema operativo, se deben tener unas fuentes del kernel compiladas, aunque basta con tener las cabeceras de esa versión del kernel. Para ello se comprueba inicialmente la versión del kernel instalada en el equipo mediante la ejecución del comando:

```
> uname -a
Linux s1 2.6.26-1-686
```

El resultado dado por el equipo es la versión del kernel junto con la fecha actual que tiene el sistema y con su arquitectura. Por tanto, teniendo la versión del kernel se procede a instalar las cabeceras del sistema ejecutando la siguiente directiva:

```
> apt-get install linux-headers-2.6.26-1-686
```

Una vez realizado este paso previo se procede a descargar de la misma página indicada en el punto anterior, las fuentes originales de la última versión de esta

aplicación. Se ha decidido guardarla en el directorio "/usr/src", donde se debe descomprimir el paquete de las fuentes ejecutando este comando:

```
> tar zxf pvfs2-2.8.1.tar.gz
```

Se obtiene un directorio con el mismo nombre que el fichero, eliminando la extensión, y dentro de éste se deben realizar diversos pasos. El primero de estos pasos es el de ejecutar el archivo de configuración del sistema de ficheros, con los argumentos pertinentes.

```
> /configure --prefix=/opt/pvfs --with-kernel=/usr/src/linux-
headers-2.6.26-1-686
```

El primero de los argumentos que se aplican en el anterior comando, sirve para indicar al instalador la ruta de destino que debe tener la aplicación, en este caso el directorio "/opt", y el segundo argumento sirve para que se configure el módulo del sistema de ficheros para la versión del kernel indicada por las cabeceras del sistema operativo presente en el equipo.

Es vital recordar que para seguir este paso y los siguientes, es absolutamente necesario tener instalado en el equipo el compilador de C, *gcc*.

El siguiente paso para la instalación del sistema de ficheros PVFS son los puntos referentes a su compilación, que debe realizarse ejecutando los comandos:

```
> make
> make install
```

Después de realizar este paso se procede a compilar e instalar los archivos necesarios para insertar en el sistema operativo el módulo del sistema de ficheros PVFS, mediante la aplicación de estos comandos:

```
> make kmod
> make kmod_install
```

Después de llevar a cabo el paso anterior, el módulo esta disponible para ser aplicado al kernel del sistema operativo, quedando almacenado en un fichero de instalación.

Para comenzar este paso se deben ejecutar las siguientes directivas:

```
> depmod -a
> modprobe pvfs2
```

Finalizado este paso, el módulo ya ha quedado cargado y listo para usarse en el sistema operativo, con la única pega de que al reiniciarse el equipo, el módulo deja de estar disponible y debe ser cargado manualmente. Para que no quede desactivado el módulo y sea cargado siempre al iniciarse el sistema, se debe modificar el archivo "/etc/modules", agregado una línea al final que ponga "pvfs2".

Terminada la carga del módulo en el sistema, se comienza la configuración del sistema de almacenamiento propiamente dicha. En primer lugar se debe generar el archivo de configuración, en el que se indican los "*I/O servers*", los "*Metadata servers*" y los directorios que serán compartidos por los servidores. Para realizar este paso se debe ejecutar la sentencia:

```
/opt/pvfs/bin/pvfs2-genconfig /etc/pvfs2-fs.conf
```

La configuración que se ha ideado para el cluster es utilizar los 4 primeros nodos de la subred saitama como servidores de E/S, que son los responsables de almacenar los datos, ya que cuentan con un disco duro añadido. Además actuarán también estos cuatro nodos como servidores de metadatos y por último utilizar los 32 nodos del cluster como clientes.

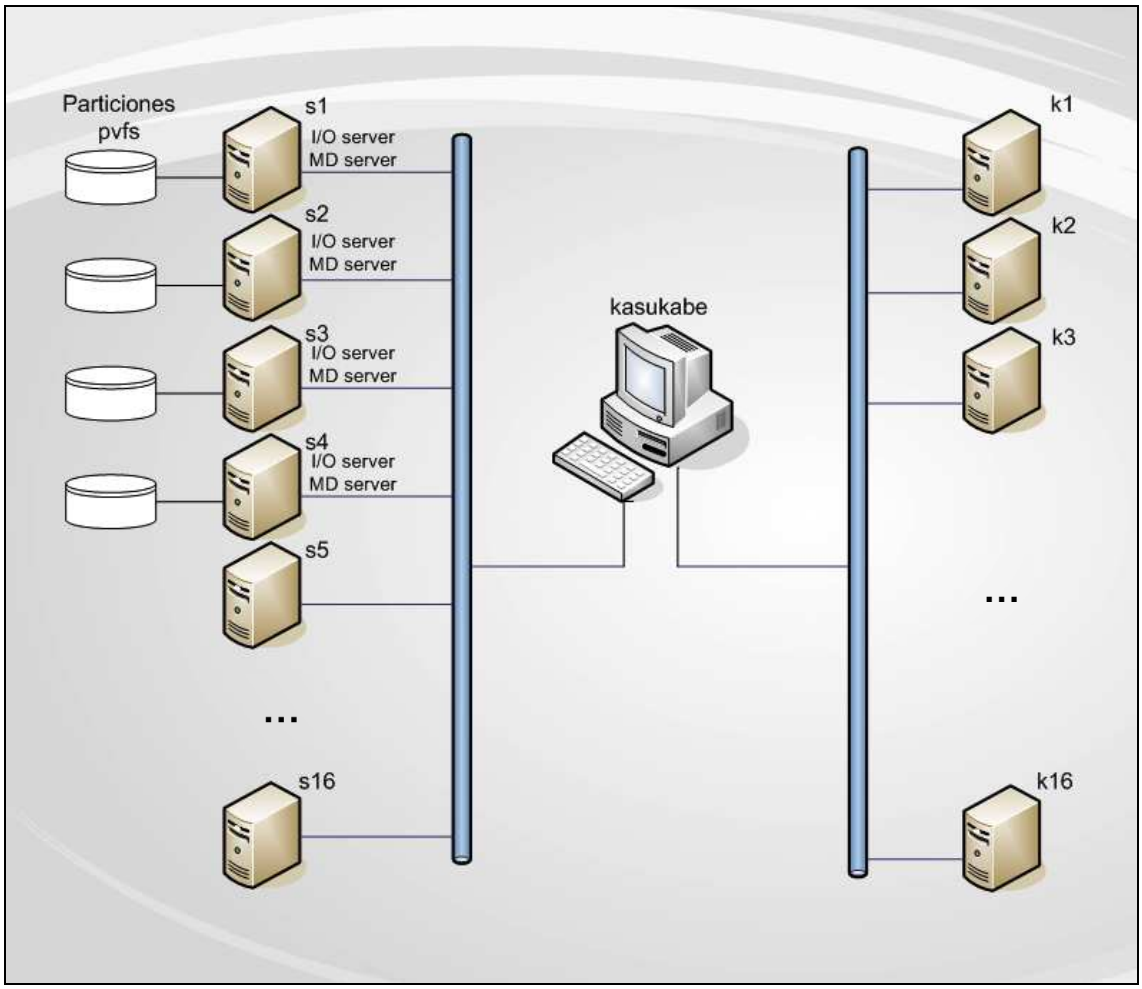

Figura 21: Arquitectura PVFS aplicada al cluster

El archivo de configuración, creado en un solo equipo del cluster, debe ser replicado en todos los demás nodos, mediante copias directas.

Para que todos los nodos queden configurados como se ha especificado anteriormente, se debe ejecutar en todos ellos las siguientes directivas referentes a los servidores.

```
/opt/pvfs/sbin/pvfs2-server /etc/pvfs2-fs.conf –f
/opt/pvfs/sbin/pvfs2-server /etc/pvfs2-fs.conf
```

Como todos los nodos del cluster van a actuar como servidores de datos, todos deberán tener creado un directorio donde se almacenará la información, que para este proyecto será "/mnt/pvfs2".

Para que todos los nodos actúen también como clientes, como se ha especificado en la organización ideada previamente, se deben ejecutar en todos ellos los siguientes comandos dentro de la carpeta "/usr/src/pvfs2/src/apps/kernel/linux":

```
> ./pvfs2-client -p ./pvfs2-client-core
> mount –t pvfs2 tcp://s1:3334/pvfs2-fs /mnt/pvfs2
```

5.2.6.1.4 Lustre

La instalación del sistema de ficheros ha sido el más costoso de entre todos los instalados para ser probados en el cluster. Inicialmente se estudiaron varios documentos de instalación y configuración que ofrece la página de *Sun Microsystems* sobre Lustre, y en todos ellos recomiendan la descarga de las fuentes del sistema junto con un kernel parcheado, o un kernel original para parchearlo a continuación.

Se intentó seguir la documentación siguiendo todas las alternativas que ofrecían pero ninguna de ellas resultaba satisfactoria, hasta que finalmente se dio con la solución definitiva gracias a la página "http://www.pdsi-scidac.org/repository/debian/", del "*Petascale data storage institute*", para descubrimientos científicos a través de la computación avanzada ("*Scientific Discovery through Advanced Computing*").

En esta página ofrecen un repositorio de paquetes para agregarlo a la lista del *apt* en "/etc/apt/sources.list" para facilitar su instalación. Estas líneas son:

```
deb http://www.pdsi-scidac.org/repository/debian testing main
deb-src http://www.pdsi-scidac.org/repository/debian testing main
```

Una vez realizado este paso se procede a actualizar la lista de paquetes ofrecidos por la aplicación *apt*, y agregar una clave de aceptación de paquetes no verificados de la página de la que se desean descargar las fuentes.

```
> apt-get update
> apt-get install pdsi-scidac-keyring
```

Terminado este proceso se puede comenzar con la instalación de las aplicaciones necesarias para el funcionamiento de lustre, junto con el kernel parcheado para lustre que ofrece dicho repositorio y sus módulos pertinentes:

```
> apt-get install linux-image-2.6.22.19-lustre-686 lustre-utils
lustre-modules-2.6.22.19-lustre-686
```

En este punto cabe destacar que el paquete indicado en la página antes mencionada, es para el kernel 2.6.18, pero se prefiere instalar la versión del kernel más actualizada que sea posible y mediante el listado de paquetes se observó que la última versión ofrecida era la 2.6.22.

Finalizada la instalación de los paquetes anteriores, se debe tener especial cuidado con las modificaciones que ha realizado en el sistema operativo que está funcionando actualmente, ya que cambia el fichero "menu.lst" alojado en el directorio "/boot/grub". Este fichero se debe reorganizar, eliminando las entradas agregadas por esta instalación de los actuales kernel que están disponibles en el equipo y quedándose solo las entradas anteriores. Además las entradas añadidas para el nuevo kernel de Lustre se modifican dejándolas como se expone a continuación:

```
title         Debian GNU/Linux, kernel 2.6.22.19-lustre-686
root          (hd0,0)
kernel        /boot/vmlinuz-2.6.22.19-lustre-686 root=/dev/hda1 ro
initrd        /boot/initrd.img-2.6.22.19-lustre-686

title         Debian GNU/Linux,kernel 2.6.22.19-lustre-686(single-user mode)
root          (hd0,0)
kernel        /boot/vmlinuz-2.6.22.19-lustre-686 root=/dev/hda1 ro single
initrd        /boot/initrd.img-2.6.22.19-lustre-686
```

A continuación se procede a configurar los servidores de metadatos (MDS/MDT) y los servidores de datos (OST/OSS).

Como se explicado en el punto 4.1.2, la estructuración que se ha seguido para este sistema de ficheros será la siguiente:

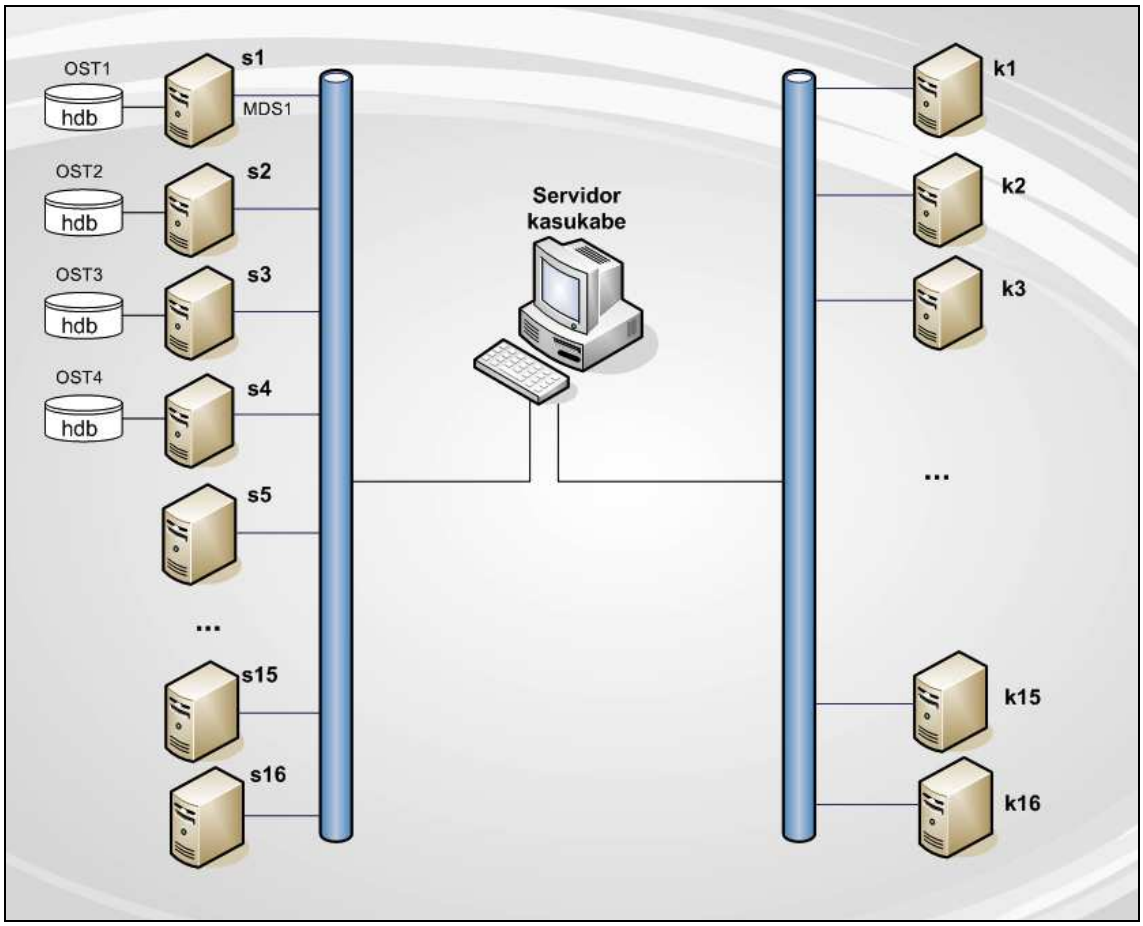

Figura 22: Arquitectura Lustre aplicada al cluster

Podemos observar como los 4 primeros nodos del subcluster saitama realizan las funciones de servidores de datos, mientras que el nodo “s1” actúa también como servidor de metadatos. El resto de los nodos del cluster actúan como clientes de Lustre y deben ser configurados para tal efecto.

La configuración de los servidores de datos y metadatos requiere de varios pasos que se deben seguir exactamente como se describe en las siguientes líneas.

El primer paso que debe ser llevado a cabo es asignarle el formato de Lustre a la partición que actuará como almacenamiento de los metadatos, que en este caso será “hda5” para el nodo “s1”, y a continuación se monta en un directorio creado para tal efecto. Se realiza ejecutando las siguientes directivas:

```
> mkfs.lustre --fsname datafs --mdt --mgs /dev/hda5
> mount -t lustre /dev/hda5 /mnt/data/mdt
```

Realizado este paso se procede a ejecutar los comandos orientados a formatear las particiones de almacenamiento de datos (OST) en el sistema de ficheros Lustre para los 4 nodos servidores de objetos (OSS) y montarlas:

```
> mkfs.lustre --fsname datafs --ost --mgsnode=s1@tcp0 /dev/hdb1
> mount -t lustre /dev/hdb1 /mnt/data/ost1/
```

Después, y solo después de haber llevado a cabo estos pasos previos, se puede comenzar a montar desde los clientes los dispositivos ofrecidos por el servidor de metadatos.

```
> mount -t lustre s1@tcp0:/datafs /mnt/data/lustre
```

5.2.6.2 Instalación del cliente Torque

En primer lugar se deben copiar los scripts de instalación que se encuentran en el directorio "/opt/torque-2.4.0b1":

- torque-package-mom-linux-i686.sh
- torque-package-pam-linux-i686.sh
- torque-package-server-linux-i686.sh

Además se debe enviar también al mismo destino la carpeta que se encuentra en el mismo directorio llamado "packages".

Se comienza la instalación propiamente dicha ejecutando los siguientes comandos en el cliente:

```
> torque-package-pam-linux-i686.sh --install
> torque-package-server-linux-i686.sh --install
> torque-package-mom-linux-i686.sh --install
```

Al terminar este punto, la aplicación quedará instalada en el cliente de forma demasiado básica y mal configurada. Para realizar una correcta configuración se deben seguir los siguientes pasos:

En primer lugar se debe modificar el fichero que se encuentra en la ruta "/var/spool/torque/server_name" escribiendo el nombre del servidor, que en este caso debe ser "kasukabe".

El segundo paso consiste en copiar el fichero "/etc/torque.conf" del servidor al mismo directorio del cliente, modificando posteriormente estas tres líneas:

```
PBS_START_SERVER=0
PBS_START_MOM=1
PBS_START_SCHED=0
```

En tercer lugar, se debe agregar la siguiente línea al final del fichero del cliente "/etc/bash.bashrc":

```
export PATH=$PATH:/opt/torque/bin:/opt/torque/sbin
```

Ya en el servidor, se deben configurar correctamente dos ficheros fundamentalmente, el primero es el que le indica a la aplicación los nodos que forman parte de su sistema de gestión y que se encuentra en la ruta "/var/spool/torque/server_priv/nodes" y que se ha incluido en los archivos Anexos a este documento. El segundo es el que define y configura las colas que van a ser gestionadas por el servidor, y que se encuentra en el directorio "/opt/torque/etc/queue.conf". También se ha incluido en el apartado de Anexos junto a este documento.

5.2.6.3 Instalación de MPICH2

La instalación de la biblioteca de alto rendimiento MPICH2 es no es muy complicada en comparación con las herramientas que se han expuesto anteriormente. Una vez que se tiene instalada esta biblioteca como se explica en el punto 2.1.5.1, en el cliente es tan fácil como comprimir o empaquetar con el comando *tar*, por ejemplo, ese directorio, donde se alojan los archivos de instalación en el servidor y copiarla en el cliente. Una vez se tiene en el cliente el archivo, se procede a descomprimirlo o desempaquetarlo, restaurando el directorio del servidor en el cliente.

Por último basta con ejecutar la siguiente directiva para que la aplicación quede instalada y funcionando en el equipo:

```
> make install
```

6 EVALUACIÓN

6.1 Evaluación de los benchmark

A continuación se expone una tabla con los sistemas de ficheros que van a ser evaluados en con estos benchmark. Se indica en la Tabla 3 el directorio propio de cada uno de los sistemas de ficheros que se van a probar:

Sistema de ficheros	Versión	Directorio
PVFS	2.8.1	/mnt/pvfs2
Lustre	1.6.7	/mnt/datos/lustre
NFS	3.0	/datos_nfs
OCFS2	1.4.1	/datos_iscsi

Tabla 3: Sistemas de ficheros y rutas

Para realizar la evaluación hemos escogido dos benchmarks de tipo E/S y uno más de computación. De los dos primeros, el denominado COLL_PERF se va a ejecutar en

varias iteraciones con 1, 2, 4, 8, 16 y 32 nodos. El otro benchmark de E/S es el BT-IO y una de sus propiedades características es que se debe ejecutar siempre con un número de nodos cuadrático, es decir, con 1, 4, 9, 16, 25 y 36 procesadores, aunque esta última prueba resultará ligeramente peor que si fuese real ya que tan solo se cuenta con 32 equipos.

Por tanto los benchmark que se han ejecutado para observar el rendimiento y la computación en el cluster se muestran en la Tabla 4.

Nombre benchmark	Tipo
Linpack	Computación
BT-IO	E/S - Computación
COLL_PERF	E/S

Tabla 4: Tipos de benchmark

6.2 Evaluación del cómputo

El benchmark Linpack es utilizado comúnmente para realizar las evaluaciones de cómputo tanto a nivel de nodo como a nivel de cluster. Sus referencias son ampliamente aceptadas en el sector de la supercomputación, hasta el punto de se genera semestralmente una lista a nivel mundial de los 500 supercomputadores que ofrecen los mejores resultados de este benchmark. Este ranking está disponible en la pagina "www.top500.org".

A nivel computacional no se puede esperar que el cluster que se ha desarrollado compita con los grandes supercomputadores que actualmente copan la lista de ésta página ya que debido al escaso número de nodos, y a los pocos medios económicos con los que se contaba no era un objetivo de este proyecto hacerse un hueco entre ellos.

El resultado obtenido después de la ejecución de este benchmark ha sido altamente satisfactorio, con una tasa máxima de 15,34 Gigaflops, bastante elevada para lo que se esperaba en un principio.

Si centramos nuestra atención en los clusters que han pertenecido a estas listas durante los últimos 15 años, se puede estar satisfecho, computacionalmente hablando, con el resultado que se ha desprendido de las pruebas realizadas sobre el cluster.

A continuación se expone una gráfica en la que se compara el nivel de computación en Gigaflops del cluster que se ha desplegado durante el desarrollo de este proyecto, con el número 500 de la lista TOP500.org de los últimos 15 años. Se ha dejado de reflejar el resultado de esta lista de los últimos 7 años ya que el número de Gigaflops, o incluso de Teraflops que ofrecen los cluster actuales, no permiten hacer una comparativa seria de nuestro cluster.

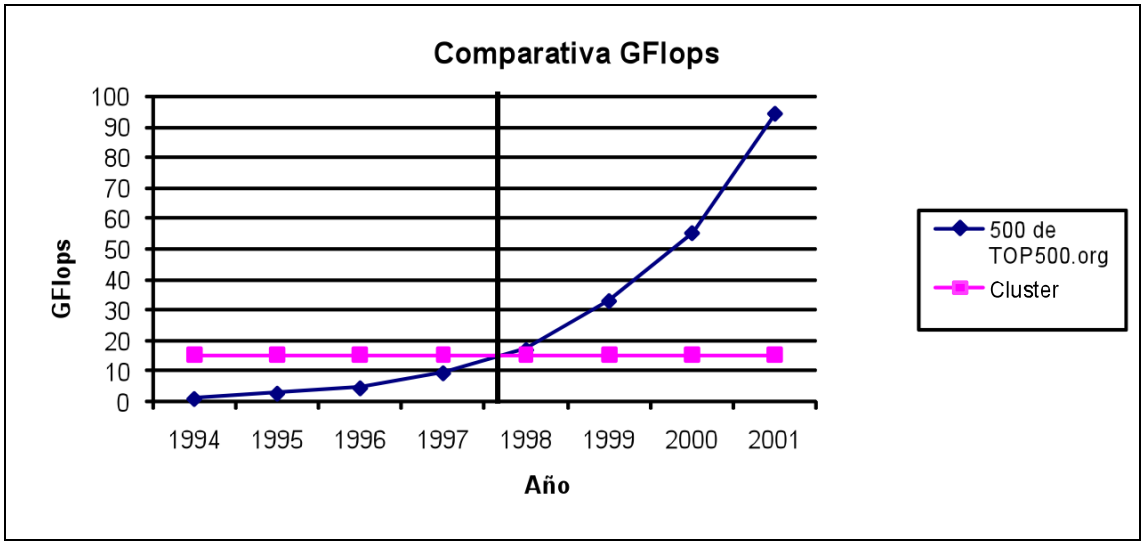

Figura 23: Comparativa computacional del cluster con último sistema del ranking TOP500.org

Se puede observar en la traza que ofrece el gráfico que el cluster que se ha construido en su mayor parte con material reciclado y mediante un gasto de lo más austero, podría haber entrado hasta principios del año 1998 dentro de la lista de los supercomputadores más potentes del planeta.

Se puede comparar también el número de equipos que se han utilizado en este cluster con el número de núcleos con los que se han obtenido los resultados del TOP500 de los mismos años con el siguiente resultado:

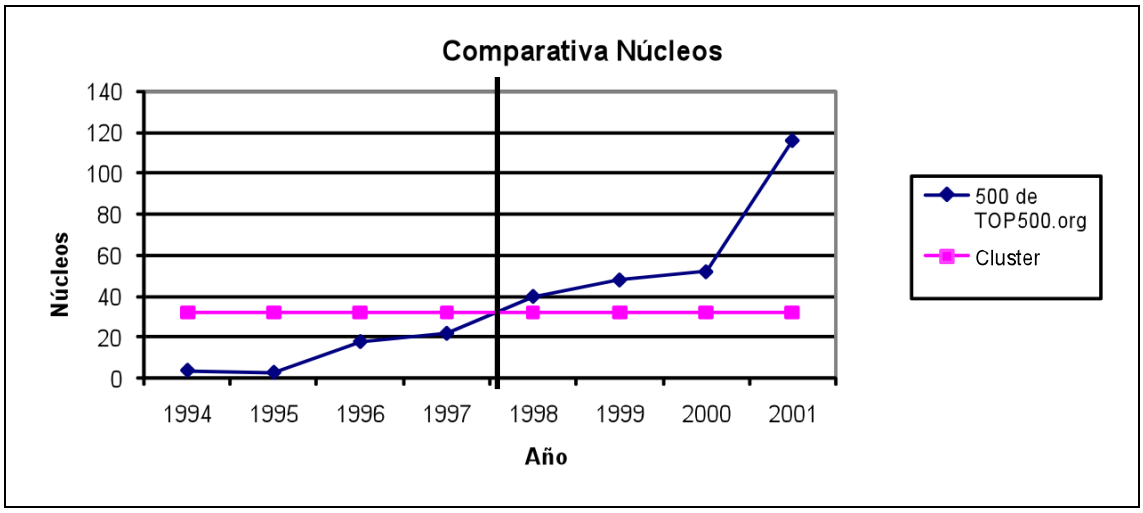

Figura 24: Comparativa en número de procesadores del cluster con el último sistema del ranking TOP500.org

La traza obtenida después de la ejecución del benchmark ofrece diversos resultados, pero el mejor de los casos para el cluster es el siguiente:

```
================================================================================
T/V                N    NB     P     Q               Time                 Gflops
--------------------------------------------------------------------------------
WR00L2L2       54000   128     2    16            6844.79              1.534e+01
--------------------------------------------------------------------------------
```

Estos datos indican, en primer lugar el algoritmo matricial que ha sido utilizado para obtener el resultado, en segundo lugar el tamaño del problema que ha sido lanzado al cluster, en tercer lugar el tamaño del bloque y el dato cuarto y quinto se corresponde con el tamaño de la matriz del problema. El tiempo que aparece en sexto lugar se corresponde con la duración de esta iteración de la prueba y por último el resultado en Gigaflops que arroja el benchmark en el cluster.

6.3 Evaluación de red

Para la evaluación de las distintas redes que conforman el cluster, se ha procedido a ejecutar el benchmark Omp 3.1.1. Con dicha evaluación se obtiene la tasa de transferencia de las distintas redes del cluster.

Se debe tener muy en cuenta antes de observar los resultados obtenidos por los benchmark de E/S la velocidad de las redes de cada uno de las dos subredes. Se recuerda que la subred kasukabe cuenta con una red de velocidad de Gbps, entre el switch y las tarjetas de red, mientras que la red saitama tiene una velocidad de 100 Mbps. Por lo tanto los nodos de kasukabe tienen una comunicación interna a una velocidad muy superior a los de saitama, y debido a este motivo se ha ejecutado el siguiente benchmark para comprobar la velocidad de la red de cada uno de las dos redes del cluster.

En este benchmark, el test de la tasa de transferencia se lleva a cabo entre un emisor y un receptor, y consiste en el envío de una cantidad fija (equivalente al tamaño de la ventana) de datos al receptor, y posteriormente se queda a la espera de una respuesta. El receptor envía la respuesta sólo después de recibir todos estos mensajes. Este proceso se repite en varias iteraciones y el ancho de banda se calcula en base al tiempo transcurrido desde el momento en el que en emisor envía el primer mensaje hasta el momento en que recibe la respuesta desde el receptor y junto con el número de bytes enviados.

Los resultados obtenidos después de la ejecución de este test son mostrados en la siguiente figura, reflejando las evaluaciones para las redes kasukabe y saitama, y para nodos entre distintas subredes.

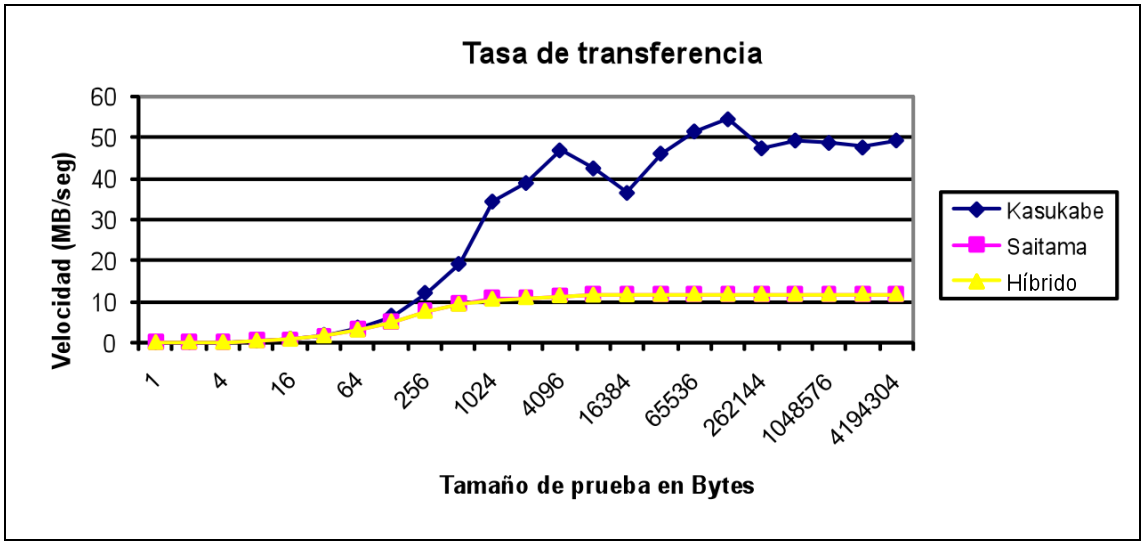

Figura 25: Gráfico de la tasa de transferencia entre las distintas redes

Como se puede observar en el gráfico anterior, la velocidad de la red existente en la red kasukabe es claramente superior a la velocidad que ofrece la red de saitama, debido al hardware del que dispone cada uno de ellas. Además, se observa como en cuanto se comunican los nodos entre las dos redes, la tasa de transferencia se adapta a la velocidad de la red más lenta, en este caso la red de saitama.

Al ser tan considerable la diferencia entre la velocidad de red de un cluster y la del otro, se ha creído conveniente realizar las pruebas de E/S cuando son con un número menor o igual a 16 nodos, en ambos cluster por separado para luego observar las diferencias entre ellos.

6.4 Evaluación de la E/S

A continuación se procede a comentar los resultados obtenidos al realizar las ejecuciones de los benchmark de E/S.

6.4.1 BTIO

El benchmark BTIO abarca muchos aspectos en lo que se refiere a la comprobación y prueba de redes, ya que además de realizar estadísticas muy fiables sobre la velocidad de lectura y escritura en un sistema de ficheros, también recoge los tiempos de envío y procesamiento que ofrecen estos sistemas cuando se está escribiendo de forma colectiva en ellos.

Las baterías de pruebas que se han lanzado sobre los sistemas de ficheros han sido, como se ha comentado anteriormente, sobre 1, 4, 9, 16, 25 y 36 procesadores, ya que una de las características principales de este benchmark es que debe ser lanzado sobre un número de procesadores cuadrático.

Además, debido a la diferencia tan considerable en las velocidades de las redes de cada una de las dos subredes, también se han realizado pruebas diferentes para los sistemas de ficheros Lustre y PVFS en uno y otro cluster. La peculiaridad de estos sistemas de ficheros es que los dispositivos físicos en los que se escriben y se leen realmente los datos se encuentran en los cuatro primeros nodos de la subred de saitama

y por tanto es interesante observar el comportamiento de unos y otros con respecto a estos sistemas de ficheros.

6.4.1.1 Evaluación sobre iSCSI

Esta prueba se ha realizado sobre el sistema de ficheros OCFS conectado entre los equipos gracias al protocolo de red iSCSI. La partición física sobre la que se accede para esta prueba se encuentra en el servidor y por tanto no se considera relevante repetir esta prueba para las dos subredes.

El resultado obtenido de la ejecución de esta prueba en el cluster es la siguiente:

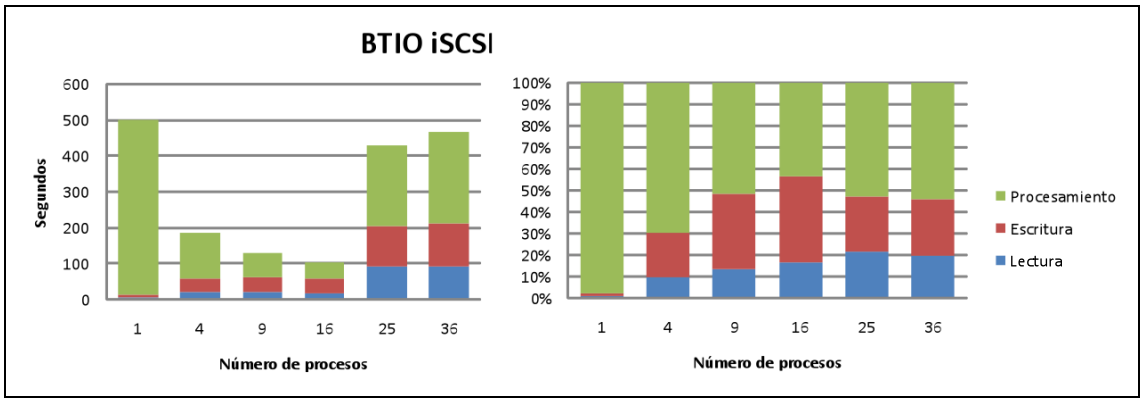

Figura 26: Resultados del benchmark BTIO sobre iSCSI

Se observa como al realizar esta prueba sobre un solo procesador, el porcentaje de procesamiento de la prueba es de prácticamente un 100% ya que al no haber conflictos con otros equipos a la hora de leer o escribir los datos, no se realiza un control constante de estos. A medida que se va ampliando el número de procesadores, el tiempo de procesamiento se reduce de forma significativa, ya que los equipos sobre los que se ha lanzado el benchmark se reparten la carga de trabajo, pero a su vez el tiempo de lectura y escritura aumentan, aunque se puede decir que de un modo muy ligero.

Finalmente los tiempos de esta prueba aumentan de forma considerable cuando se lanzan sobre 25 y 36 nodos, debido principalmente a las comunicaciones que deben producirse entre los nodos de las dos subredes (BTIO utiliza llamadas colectivas de MPI, lo que implica una comunicación entre todos los procesos involucrados). Estos nodos se comunican de forma única mediante la conexión que cada una de las subredes enlaza con el servidor del cluster, lo que seguramente pueda provocar un cuello de botella.

6.4.1.2 Evaluación sobre NFS

La segunda prueba lanzada sobre el cluster ha sido realizada para observar el rendimiento del sistema de ficheros NFS. Al igual que en el caso de iSCSI, el dispositivo que se exporta realmente a todos los nodos del cluster se encuentra físicamente en el servidor y por tanto no se ha considerado necesaria la repetición de esta prueba en una parte del cluster y en la otra.

El resultado obtenido por este benchmark es el siguiente:

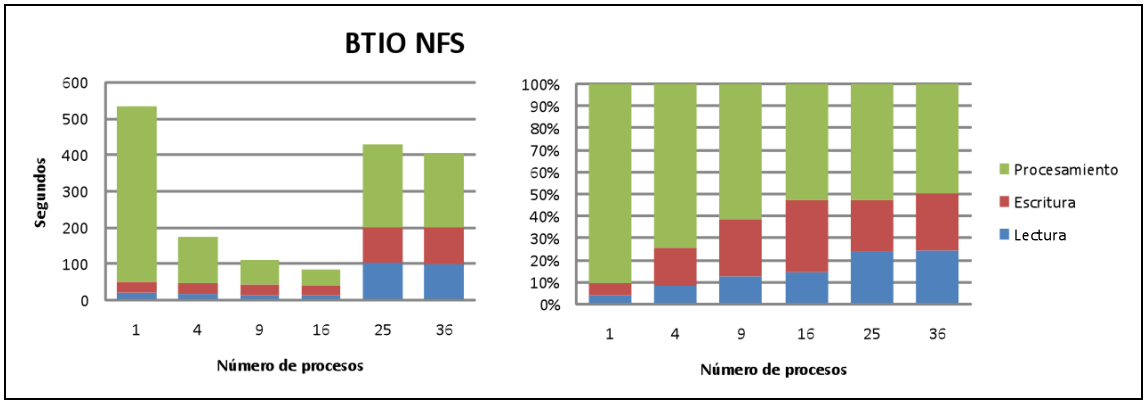

Figura 27: Resultados del benchmark BTIO sobre NFS

Los resultados para este sistema de ficheros se puede observar como son muy similares a los arrojados por la ejecución de este mismo benchmark sobre iSCSI. Existe una ligera diferencia entre los dos resultados, pero que se puede apreciar significativamente al ejecutar la prueba con un solo nodo. Se observa como los tiempos de procesamiento son casi idénticos, pero mientras que en el sistema de ficheros OCFS2 los tiempos de lectura y escritura son casi nulos, en NFS se puede considerar que estos tiempos abarcan casi un 10% de la ejecución.

Por otro lado, el aumento de procesadores en esta prueba provoca un descenso más acusado en el tiempo de ejecución que en el sistema de ficheros anterior, aunque tampoco se puede concluir que la ventaja temporal sea un factor decisivo a la hora de escoger un sistema u otro.

6.4.1.3 Evaluación sobre PVFS

Esta prueba se ha realizado sobre el dispositivo compartido en PVFS. Como se ha explicado anteriormente, este dispositivo compartido esta formado por un conjunto de particiones de los discos duros añadidos a los cuatro primeros nodos de la subred

saitama. Es por ello que en este caso si que se ha considerado interesante observar los resultados que se obtienen mediante la ejecución de este benchmark tanto en la subred kasukabe como en saitama.

El resultado de esta evaluación para la red kasukabe es el siguiente:

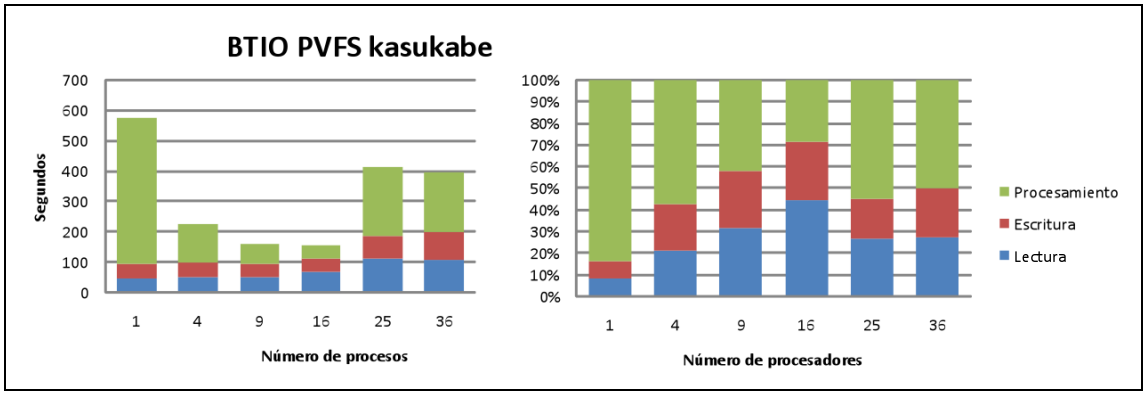

Figura 28: Resultados del benchmark BTIO sobre PVFS en kasukabe

Se observa en este gráfico como el benchmark para este sistema de ficheros en concreto, escala perfectamente viendo como hasta los resultados obtenidos con los 16 nodos el tiempo gastado, en términos absolutos, en procesamiento es de menos del 30%, mientras que en las pruebas anteriores el tiempo de procesamiento estaba más cerca del 50%.

La mayor diferencia que se encuentra en estos resultados en comparación con los anteriores, es que todos los tiempos de ejecución de la prueba son visiblemente superiores tanto a los de NFS como a los de iSCSI. Es debido principalmente a que las comunicaciones de estos nodos con los servidores de metadatos solo pueden ir por un cable (el que conecta cada una de las dos subredes con el servidor) lo que provoca un cuello de botella considerable.

El resultado obtenido para este sistema de ficheros en la subred saitama es el siguiente:

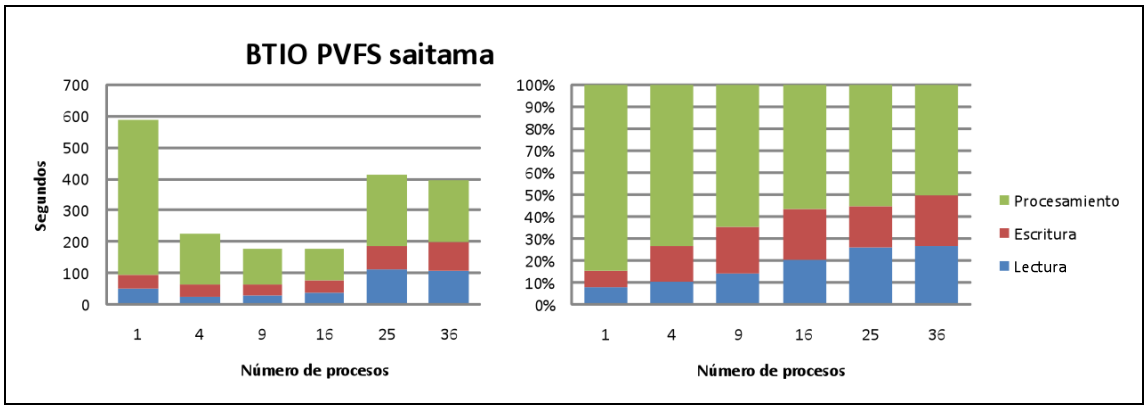

Figura 29: Resultado del benchmark BTIO sobre PVFS en saitama

El resultado obtenido es ligeramente peor que el que se podía esperar en un principio ya que se consideraba que al pertenecer los nodos que ejecutan esta prueba a la misma subred donde están ubicados los servidores tanto de datos como de metadatos, las comunicaciones se producirían de forma más fluida. Aun así, este benchmark reproduce llamadas colectivas, lo que supone una comunicación constante entre los nodos que están ejecutando la prueba y por tanto la velocidad de comunicación entre estos nodos también juega un papel fundamental a la hora de obtener esos tiempos.

A pesar de que los tiempos totales son ligeramente superiores a los obtenidos por el cluster kasukabe, si se separan los tiempos de lectura y escritura del resto, se observa como realmente estos tiempos se reducen de forma más visible en la subred saitama. Como se ha comentado en el punto anterior, lo que hace que esta subred sea más ineficiente son los tiempos de comunicación entre los nodos.

6.4.1.4 Evaluación sobre Lustre

Al igual que en el caso anterior, este sistema de ficheros tiene como peculiaridad que el dispositivo que se comparte entre los nodos del cluster está formado por particiones pertenecientes a los cuatro discos duros instalados en los primeros nodos de la subred saitama. Por tanto al igual que en el sistema de ficheros PVFS se han realizado las pruebas para cada uno de las dos subredes por separado.

En primer lugar, el resultado obtenido por este benchmark en kasukabe es el siguiente:

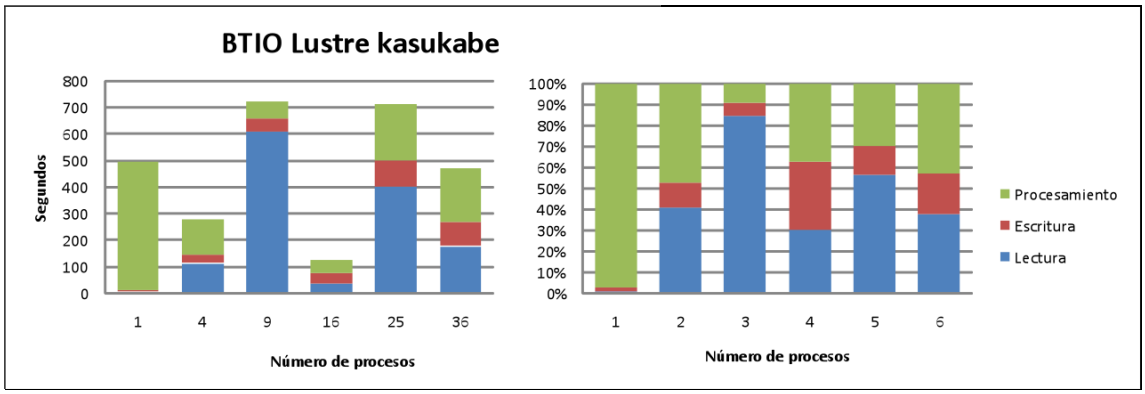

Figura 30: Resultados del benchmark BTIO sobre Lustre para kasukabe

Como se muestra en la Figura 30, Lustre presenta problemas de accesos con 9 procesos. El resto de pruebas no presentan esta anomalía. Para los casos de 25 y 36 procesos, el tiempo de lectura se incrementa sustancialmente frente al tiempo de escrituras.

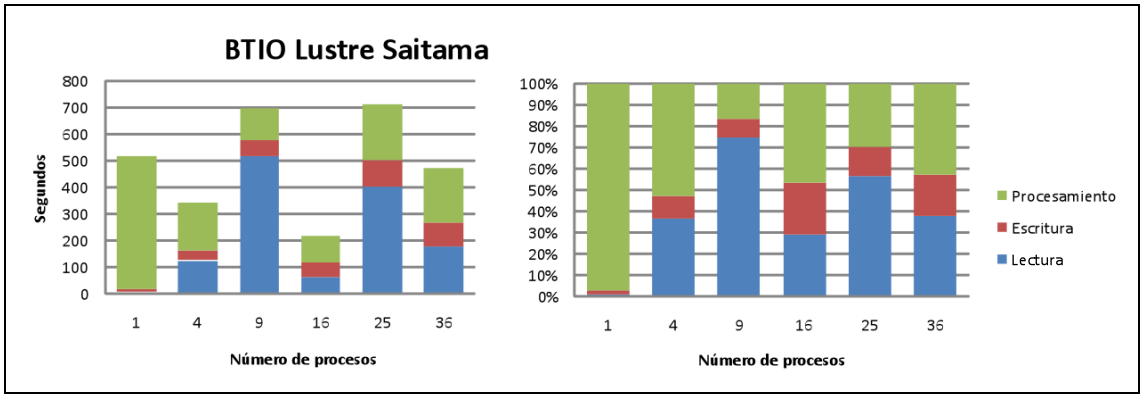

Figura 31: Resultado para el benchmark BTIO sobre Lustre en saitama

Al igual que en los casos anteriores, en la Figura 31 se muestran los resultados para la red saitama. Como se puede observar, no se aprecia mejora alguna. Esto es debido a que los servidores de datos se encuentran dentro de la subred más lenta. En trabajos futuros se evaluará el comportamiento de este sistema de ficheros en la red más rápida.

Con el objetivo de investigar qué está pasando en el caso de 9 procesos, se ha procedido a usar otra evaluación. En está evaluación se estudiará el comportamiento del

sistema de ficheros Lustre para 9 procesos. Para llevar a cabo este objetivo se ha utilizado otro benchmark llamado SimParIO.

SimParI/O es un benchmark de utilizado para evaluar distintos sistemas de E/S. El comportamiento de esta aplicación es el siguiente: primero se crea el fichero de pruebas, segundo se procede con 20 escrituras no solapadas, a continuación se realizan 20 lecturas no solapadas, y por último se cierra el fichero. Se han obtenido resultados para distintos tamaños de acceso.

Los resultados obtenidos demuestran que le problema persiste solo para los accesos de lectura, por lo que no se procederá a mostrar los resultados para las escrituras. Los resultados de está evaluación se muestran en la Figura 32:

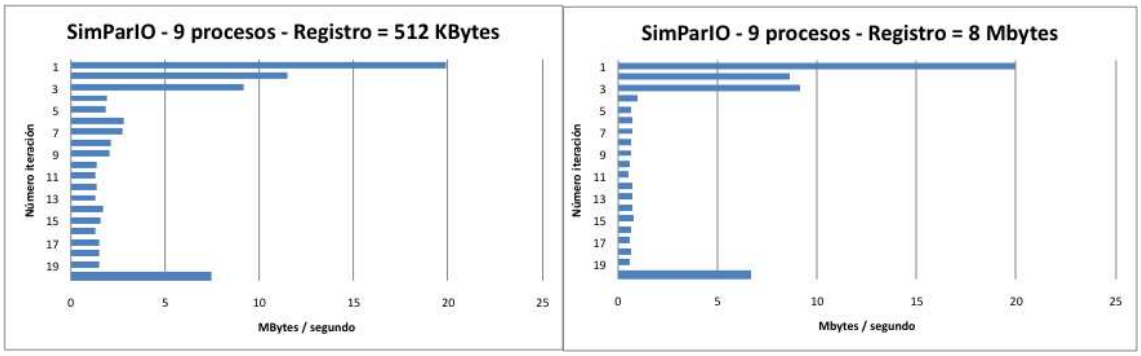

Figura 32. Histogramas de Lustre sobre 9 procesos

Con los resultados mostrados anteriormente, se puede concluir que el problema aparece para todos los tamaños de acceso, (incluso para pequeños). Se han obtenidos tiempos para tamaños de acceso iguales a BTIO. Tras las distintas evaluaciones, no se ha podido averiguar la causa de esté problema. Quizás el problema se debe a la granularidad generada al acceder con 9 procesos. Como trabajo futuro se propone evaluar y averiguar la causa de esta perdida de rendimiento.

6.4.1.5 Comparativa de los sistemas de ficheros

A continuación se expone una pequeña comparativa con los resultados obtenidos para cada uno de los sistemas de ficheros probados con este benchmark, realizando una gráfica de los tiempos que tarda cada uno de estos sistemas de ficheros tanto en la lectura como en la escritura de datos.

Para las lecturas de datos se han obtenido los siguientes resultados:

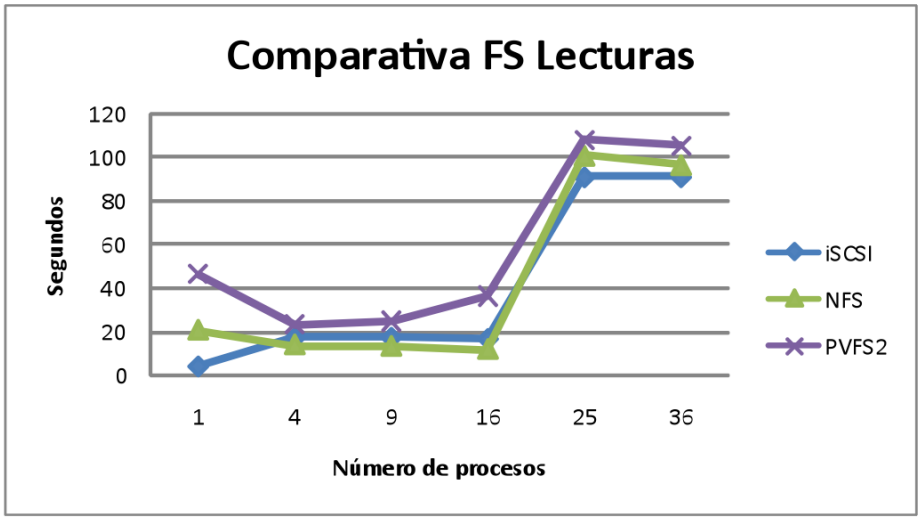

Figura 33: Comparativa de lecturas con BTIO

A la vista de los resultados, la mejor opción para las operaciones de lectura es NFS. Esto es debido a que no se ve afectado por ningún mecanismo de bloqueo (locking). Para cualquier número de procesos, PVFS2 resulta ser la peor opción, debido a que los servidores de datos se encuentran en la red más lenta. En la figura se ha prescindido representar Lustre debido a los motivos comentados en la subsección anterior.

Para las escrituras, el grafico comparativo obtenido es el siguiente:

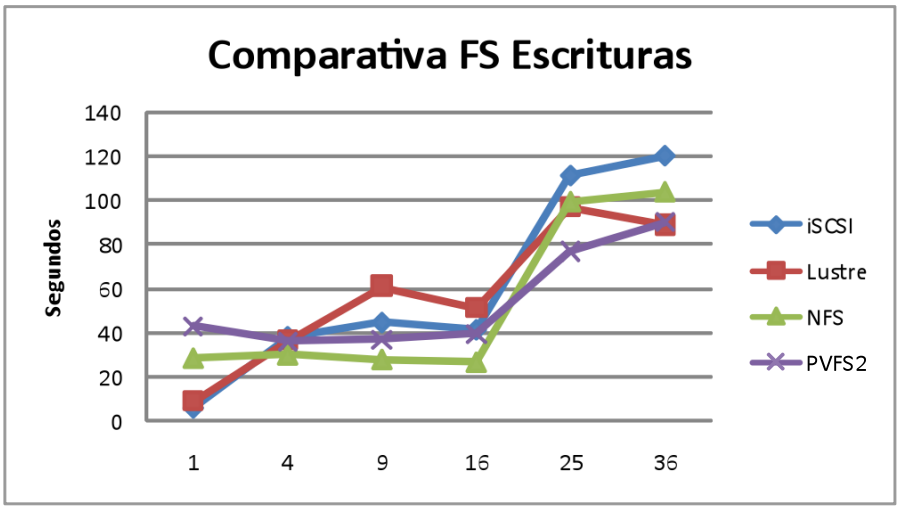

Figura 34: Comparativa de escrituras con BTIO

En el caso de las escrituras, se puede observar como la evolución en los cuatro sistemas de ficheros sigue una línea casi idéntica, exceptuando el sistema de ficheros Lustre, que como se puede observar a medida que crece el número de procesadores

tiene un comportamiento bastante irregular y crece significativamente el tiempo de ejecución de las escrituras. El resto de sistemas de ficheros siguen un comportamiento regular, manteniendo el mismo tiempo de ejecución para las escrituras, hasta que empiezan a trabajar juntas las dos subredes (a partir de 16 nodos) que se incrementa notablemente el tiempo de ejecución debido a las comunicaciones entre los nodos. En conclusión, la mejor opción cuando el número de nodos aumenta es Lustre.

6.4.2 COLL_PERF

A continuación se exponen y se comentan los resultados obtenidos después de ejecutar el benchmark "Coll_perf". Este benchmark se centra en obtener los tiempos tanto de escritura como de lectura para los diferentes sistemas de ficheros para los que se lanza.

Al igual que para el benchmark anterior, las pruebas sobre los sistemas de ficheros Lustre y PVFS han sido repetidas para cada uno de las dos subredes con el fin de obtener diferentes resultados y realizar con ellos comparativas concluyentes.

Este benchmark puede ser lanzado en cualquier número de procesos y por tanto se ha considerado suficientemente significativo lanzar la prueba sobre 1, 2, 4, 8, 16 y 32 nodos. Además los resultados que se han generado a lo largo de todas las pruebas de este benchmark han sido modificados para obtener el dato "Tasa de transferencia", que se ha conseguido mediante el cálculo:

Tasa transferencia = Tamaño de prueba (MBytes)/ Tiempo (segundos)

Para este benchmark se han obtenido resultados de la escritura y lectura de datos diferentes tamaños, de 64 MBytes, de 128 MBytes y de 512 MBytes, pero a la hora de realizar las gráficas se ha considerado que las tasas más significativas obtenidas han sido las conseguidas por la ejecución de la última prueba. Por tanto todas las graficas expuestas en este punto han sido generadas a partir de la prueba realizada con el problema de tamaño 512 MBytes.

6.4.2.1 Evaluación sobre iSCSI

Esta prueba ha sido realizada sobre el dispositivo formateado en el sistema de ficheros OCFS, interconectado mediante el protocolo iSCSI, como en el caso del benchmark anterior.

Los resultados obtenidos de esta prueba en el cluster han permitido generar la siguiente figura:

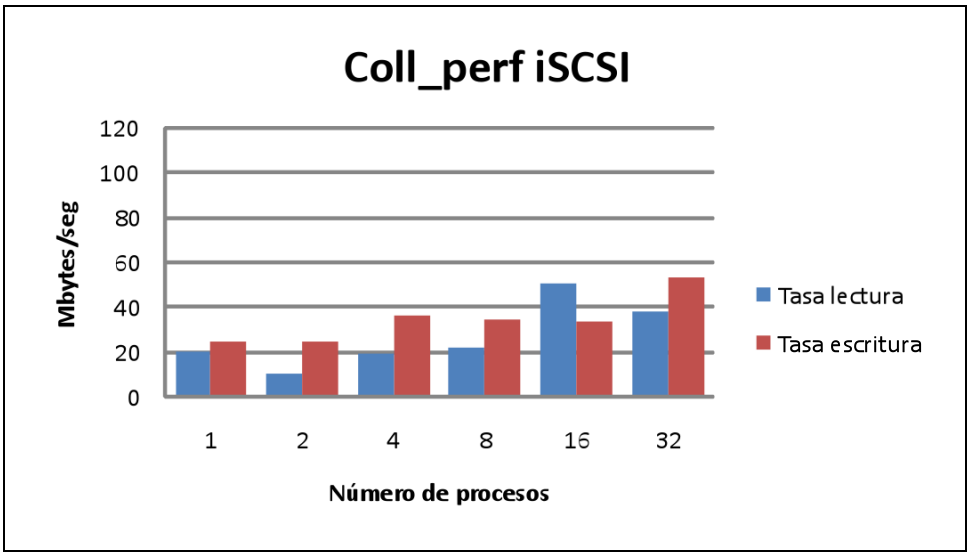

Figura 35: Resultados del benchmark Coll_perf en iSCSI

En este caso, el resultado crece de forma considerable cuantos más nodos se incluyan en la prueba. Además se observa como la tasa de escritura es superior en la mayoría de los casos a la tasa de lectura con lo que en principio se puede concluir que este benchmark tiene más facilidad de acceso al dispositivo cuando tiene que escribir que cuando quiere leer del disco.

6.4.2.2 Evaluación sobre NFS

La segunda prueba que se ha realizado ha sido sobre el dispositivo compartido desde el servidor mediante el protocolo de red NFS. Por esta razón, y al igual que en el caso de iSCSI, se considera que realizar esta prueba sobre las dos subredes no es necesario.

Los resultados obtenidos de la ejecución de este benchmark han sido los que se pueden observar en la Figura 36.

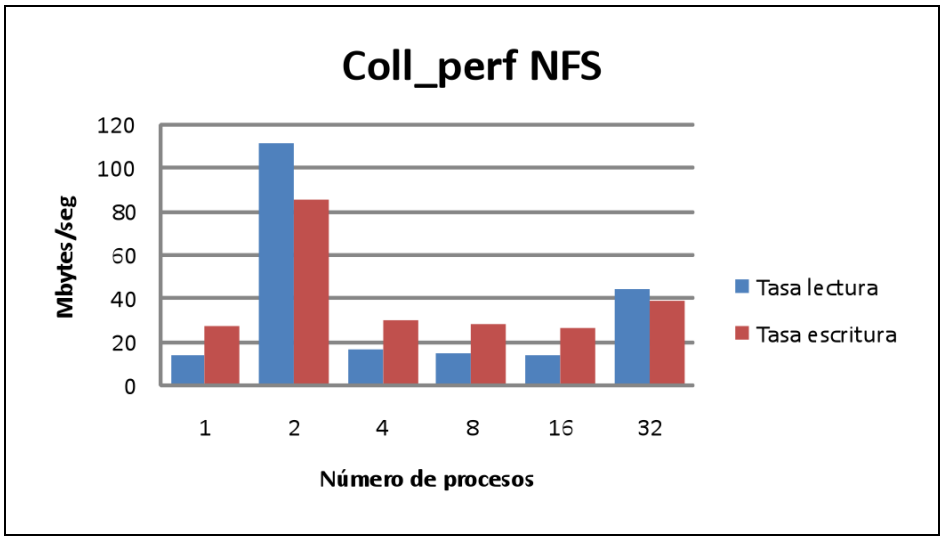

Figura 36: Resultados del benchmark Coll_perf sobre NFS

En la Figura 36 se observa que para 2 procesos NFS tiene un comportamiento no esperado. Aparentemente no se encunentra una explicación a primera vista.

6.4.2.3 Evaluación sobre PVFS

Como ya se ha comentado en los puntos anteriores, al ser un dispositivo compuesto por varios discos duros, y estar todos ellos en la subred saitama, se cree interesante realizar esta prueba por separado tanto en la subred de saitama como en la subred kasukabe.

En primer lugar, se ofrecen los resultados obtenidos después de lanzar este benchmark en la subred kasukabe.

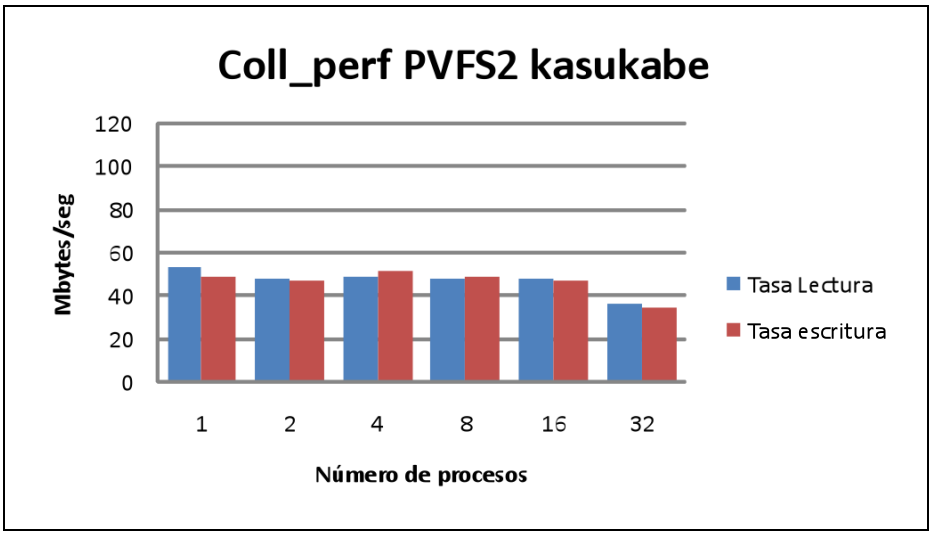

Figura 37: Resultados del benchmark Coll_perf sobre PVFS en kasukabe

El resultado obtenido por este benchmark en el sistema de ficheros PVFS ha sido altamente satisfactorio ya que, comparando estos datos con los obtenidos sobre los sistemas de ficheros anteriores se produce una mejoría de casi el doble de la tasa de transferencia en muchos de los datos obtenidos.

Este resultado puede significar que el sistema de ficheros PVFS ofrece un mejor rendimiento en lo que se refiere a lecturas y escrituras del que puedan ofrecer sistemas como NFS u OCFS, al menos en redes de velocidad gigabit.

El resultado obtenido en la subred saitama es el siguiente:

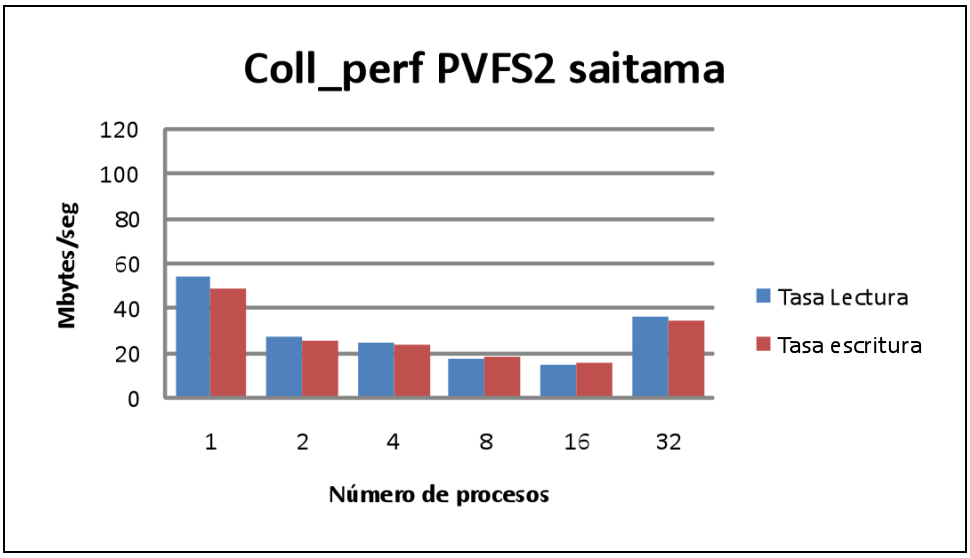

Figura 38: Resultado del benchmark Coll_perf sobre PVFS en saitama

Mientras que el resultado obtenido en la subred anterior ha sido ampliamente satisfactorio, el resultado que arroja este sistema de ficheros para una red de menor velocidad, no se puede considerar que mejor excesivamente los datos obtenidos al utilizarse otros sistemas de ficheros. Incluso se puede afirmar que esta tecnología, en una red que no ofrezca altas prestaciones de velocidad, no merece la pena ser utilizada por delante de otros sistemas de ficheros más sencillos como NFS.

6.4.2.4 Evaluación sobre Lustre

Al igual que en el caso de PVFS, el dispositivo que se comparte mediante el sistema de ficheros Lustre está compuesto por cuatro discos duros repartidos en los cuatro primeros nodos de la subred de saitama. Por tanto para realizar una comparativa

interesante del comportamiento de este sistema de ficheros se ha realizado sobre la subred saitama y sobre la subred kasukabe por separado.

En primer lugar se exponen los resultados obtenidos por el benchmark en la subred kasukabe:

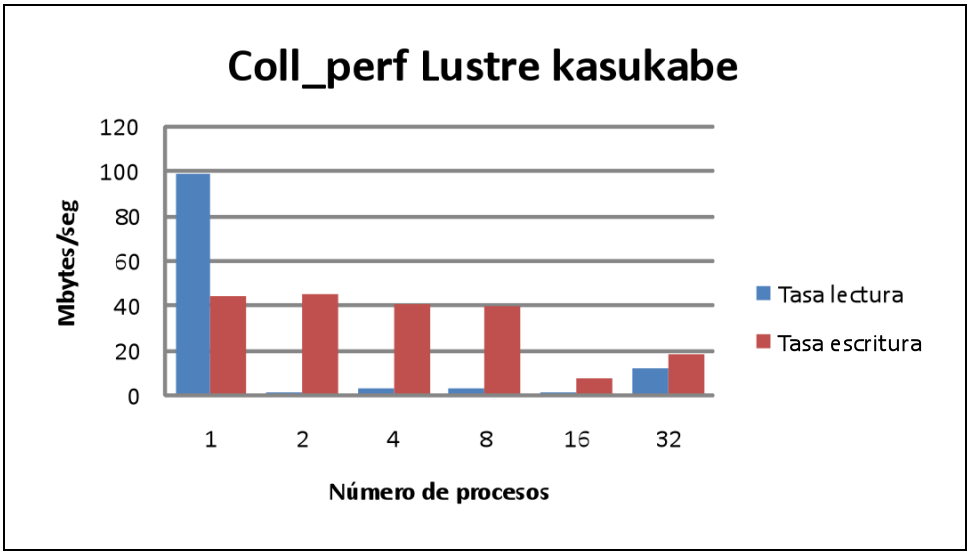

Figura 39: Resultados del benchmark Coll_perf sobre Lustre en kasukabe

Los resultados obtenidos al realizar esta prueba sobre la subred kasukabe muestran un comportamiento muy irregular de Lustre. Como se puede observar a la vista de la figura, exceptuando la prueba realizada sobre un solo nodo, en el resto la tasa de escritura es muy alta en comparación con la tasa de lectura. Además la prueba realizada sobre 16 nodos ofrece resultados muy inferiores comparados con PVFS2.

A continuación se expone el resultado obtenido de lanzar este mismo benchmark sobre la subred de saitama:

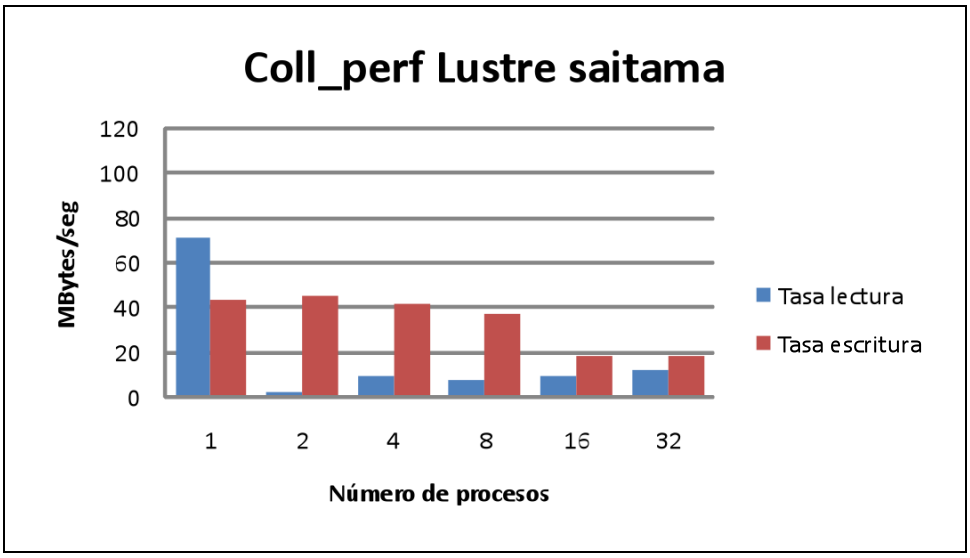

Figura 40: Resultados del benchmark Coll_perf sobre Lustre en saitama

Se observa como la situación de los dispositivos físicos en esta evaluación influye de forma considerable, especialmente a las lecturas. En este caso se puede decir que esta tasa se mantiene de manera prácticamente constante, con un ligero aumento a medida que crece el número de procesadores sobre los que son lanzadas las pruebas.

6.4.2.5 Comparativa de los sistemas de ficheros

En este punto se realiza una breve comparación de los datos obtenidos después del lanzamiento de este benchmark. Para llevar a cabo dicha comparación se mostrarán dos gráficas en las que se podrá visualizar la tasa de transferencia tanto para lecturas como para escrituras obtenidas de cada uno de los sistemas de ficheros.

Los datos obtenidos a partir de las lecturas en el cluster han dado lugar a la siguiente figura:

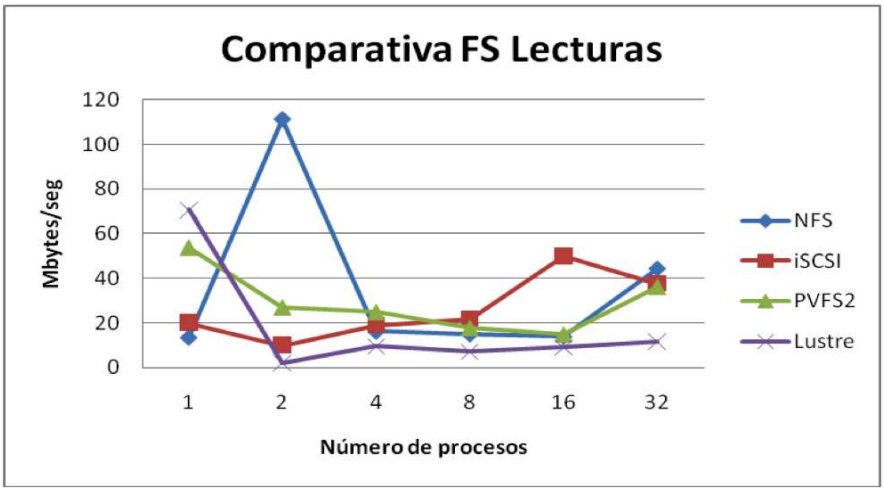

Figura 41: Comparativa de lecturas con Coll_perf

Se muestra en la Figura 41cómo de un modo más constante, tanto iSCSI como PVFS2 ofrecen una tasa de transferencia superior a la ofrecida por los otros dos sistemas de ficheros. Se puede observar también cómo para un número más pequeño de procesos el sistema que mejores resultados arroja es PVFS2 mientras que cuando el numero de procesos aumenta el sistema de ficheros más recomendable es iSCSI.

A continuación, se muestra en la Figura 42 la comparativa de escrituras de los cuatro sistemas de ficheros que se están evaluando.

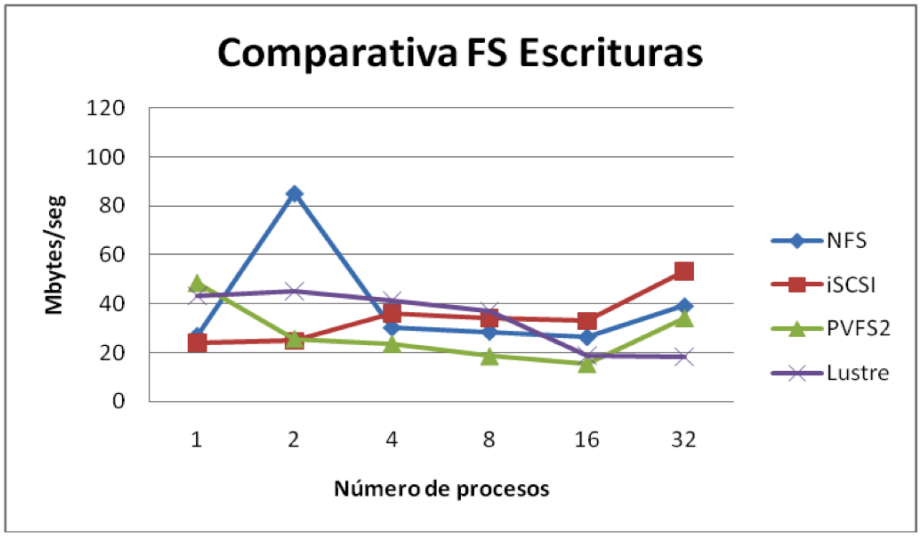

Figura 42: Comparativa de escrituras con Coll_perf

A la vista de esta figura se puede observar cómo para un número reducido de procesos sobre los que lanzar el benchmark, el sistema de ficheros que mejores resultados ofrece es Lustre. Pero a medida que el número de procesos sobre el que se ejecuta la evaluación aumenta, el sistema de ficheros Lustre pierde tasa de transferencia lentamente hasta situarse como el peor de los cuatro. Al contrario que Lustre, el sistema de ficheros que más mejora cuantos más procesos estén ejecutándose es iSCSI, erigiéndose como el sistema de ficheros más completo para las ejecuciones de este benchmark.

6.5 Conclusiones finales

Se puede concluir que las comunicaciones internas entre los trabajos que deben realizarse de forma paralela ralentizan de forma notable la tasa de transferencia que puede llegar a obtenerse, a la vista del gran escalón que se produce cuando se ejecuta para un solo. Según la evaluación de E/S, la opción más recomendable es PVFS2 sobre todas las demás. Lustre presenta un comportamiento bastante irregular. Tras los resultados obtenidos es posible que haya un problema de instalación o configuración. La documentación revisada no sugiere ninguna solución.

7 CONCLUSIONES Y TRABAJOS FUTUROS

En este apartado se exponen los objetivos que se marcaron al principio del proyecto y el grado de cumplimiento de los mismos y a continuación, se proponen futuras mejoras.

7.1 Conclusiones

En este apartado se procede a evaluar la consecución de cada uno de los objetivos marcados para la elaboración de este proyecto. No se ha podido considerar como acabado hasta que se ha comprobado que todos los objetivos marcados han sido completados:

Para cumplir con el objetivo principal de este proyecto se ha utilizado en su mayoría material que ya era utilizado anteriormente para otras labores del departamento, tal como antiguos ordenadores de las aulas del Laboratorio y un ordenador que antes hacía las funciones de servidor de dichas aulas. Entre los materiales nuevos destacan nuevas tarjetas de red a velocidad de Gigabit y un switch que también trabaja la misma velocidad, para poder poner en funcionamiento una parte del cluster a alta velocidad.

Se exponen a continuación los pros y los contras que se han considerado después de la finalización de este proyecto:

Pros	Contras
Ahorro de dinero en material	Horas de mantenimiento
Disponibilidad del cluster	Dinero en mantenimiento
Reciclado de equipos	Potencia de cómputo limitada
Facilidad de gestión	

Tabla 5: Ventajas y desventajas del proyecto

Se puede observar como el número de ventajas ofrecidas por el cluster supera con creces el número de inconvenientes encontrados, y por lo tanto se considera que va a resultar beneficioso para el Laboratorio contar con este tipo de despliegue tecnológico a su disposición.

Además se consideran cumplidos los objetivos marcados al comienzo de este proyecto:

- Se han conseguido instalar con éxito todos los sistemas de ficheros que se deseaban probar durante el proceso de evaluación de este proyecto.
- Se ha ofrecido a los usuarios del cluster la posibilidad de acceder a los datos de sus cuentas del Laboratorio del Departamento de Informática, gracias a la correcta instalación y configuración de la tecnología NIS.
- Se ha dado la posibilidad de acceder de forma remota desde cualquier equipo al servidor del cluster, mediante la tecnología de seguridad *ssh*, siempre que se posea cuenta en las aulas del Laboratorio del Departamento.
- Se han desarrollado diversos scripts de administración y se han utilizado varias aplicaciones de gestión, para facilitar el manejo del cluster tanto a los administradores como a los usuarios.
- Se ha instalado y configurado correctamente en el cluster la herramienta de gestión de colas de trabajo Torque, gracias a la cual se ha conseguido automatizar el lanzamiento de trabajos al cluster por parte de alumnos y

profesores, gestionando prioridades y tiempos de ejecución de los trabajos enviados.

Por último cabe destacar que en el transcurso del segundo cuatrimestre del curso académico 2008/2009, el cluster diseñado y desplegado ha sido utilizado exitosamente como soporte a la docencia de la asignatura Arquitectura de Computadores II.

7.2 Trabajos futuros

Para ampliar las posibilidades que puede ofrecer el presente cluster se proponen las siguientes tareas:

- La creación de una Web mediante la que se podría gestionar de forma remota el envío de trabajos y la recepción de los resultados que devuelven. Además podría contar con un sistema de monitorización, como por ejemplo Ganglia, para visualizar y gestionar la carga de trabajo de cada uno de los nodos del cluster en tiempo real.

- Realizar otro tipo de evaluaciones sobre el cluster utilizando las máquinas virtuales, con el propósito de buscar el máximo rendimiento que puedan ofrecer éstas, con sus consabidas limitaciones precisamente por ser virtuales, y comparar estos resultados con los obtenidos en las máquinas físicas.

- Ampliar la seguridad en el servidor y en el cluster, restringiendo ejecuciones de aplicaciones peligrosas y avisando mediante el envío de correos electrónicos al administrador, de la cuenta de la persona atacante y la acción que esta intentando realizar. Además sería interesante gestionar el control de acceso dependiendo del tipo de usuario; esta característica ya está incluida en *Torque*, simplemente sería necesario configurarla.

- Reorganizar el cluster para añadirlo a una arquitectura *grid*, con el objetivo de utilizar de forma coordinada todo tipo de recursos, ya sean de cómputo o de almacenamiento por ejemplo, y de forma que no se tenga sobre ellos un control centralizado. Además esta tecnología admitiría la

inclusión de nuevos nodos, sin importar sus características y arquitectura, ya que trabaja indistintamente con recursos homogéneos y heterogéneos.

- Ampliar las prestaciones de los nodos, como la inclusión de una tarjeta gráfica de altas prestaciones y un monitor para cada uno de los nodos, con el objetivo de crear un cluster de visualización. En el cual cada uno de los nodos controlaría un monitor, pero a su vez las imágenes que salen de una pantalla, entran en la siguiente de forma automática, y por tanto entra a trabajar el siguiente nodo.

7.3 Método de trabajo y presupuestos

En esta sección se describe la metodología utilizada en el proyecto así como la organización y planificación de tareas que se va a llevar a cabo en el proyecto; además, se especifican tanto los recursos humanos y materiales como la tecnología necesaria para el éxito del proyecto, y para finalizar, se da un presupuesto que refleja el coste total que supondrá el desarrollo del proyecto.

7.3.1 Método de trabajo

Para el desarrollo del proyecto se ha utilizado el ciclo de vida incremental o de versiones sucesivas. La elección de este ciclo de vida se debe además de porque el proyecto es de gran tamaño, porque algunas de las funcionalidades del proyecto deben estar operativas antes que otras y además, este método permite validar el sistema a medida que se construye.

Con este método, cada versión o fase es un sistema funcional capaz de realizar progresivamente la función deseada.

Como se puede observar en la siguiente figura, el ciclo de vida incremental posee una fase de análisis inicial y varias fases de diseño, implementación y evaluación de las pruebas de forma incremental.

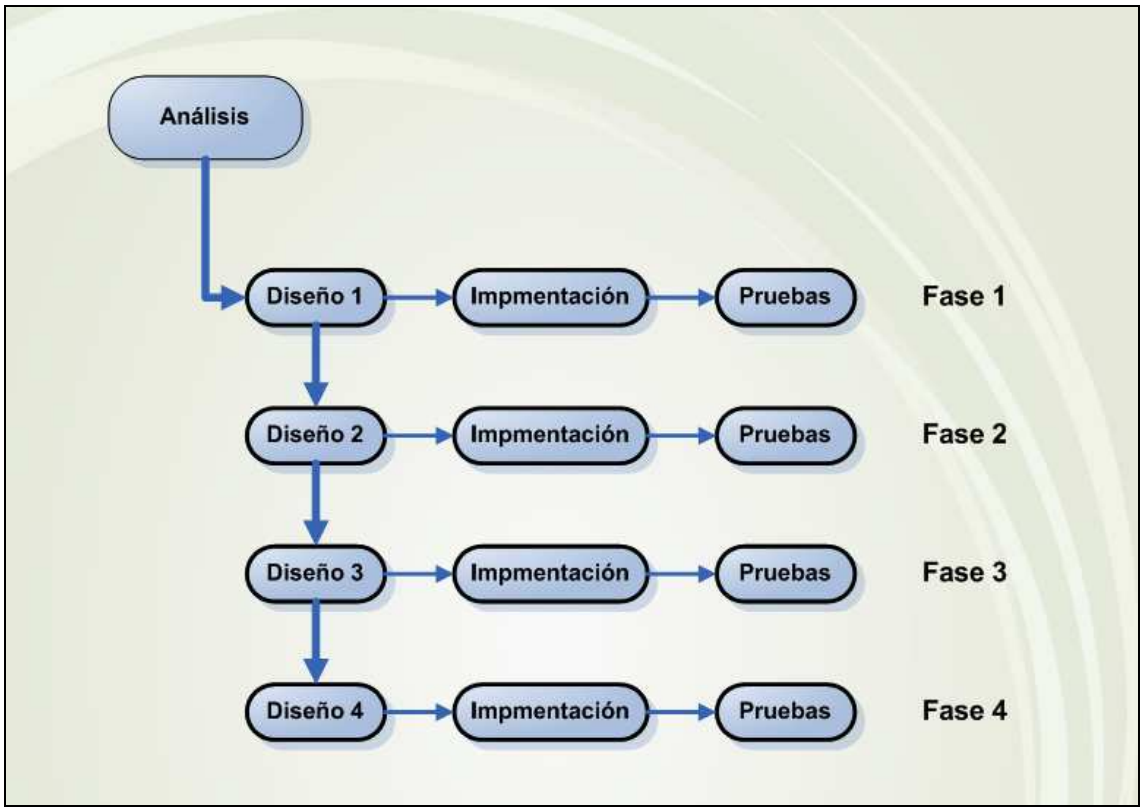

Figura 43: Ciclo de vida incremental

A continuación se resumen los objetivos de cada una de las fases del diseño incremental.

Análisis:

Se realiza un estudio de las posibles arquitecturas disponibles que pueden ser desplegadas a raíz del material hardware disponible para el desarrollo de este proyecto.

Fase 1:

Una vez realizado un estudio previo se lleva a cabo el despliegue de la arquitectura seleccionada, salvando los inconvenientes que vayan surgiendo a medida que toma forma dicho despliegue.

Fase 2:

Una vez colocado el cluster como se desea, se procede a la instalación del software necesario para el correcto funcionamiento del cluster y para dotarlo de los servicios requeridos. Además para completar estas funcionalidades se deben realizar

múltiples programas y scripts para facilitar el manejo de los servicios y de los nodos del cluster.

Fase 3:

Se procede a la ejecución de las pruebas que se han considerado de relevancia, almacenando los datos obtenidos en una organización de carpetas representativas.

Fase 4:

Se realiza una evaluación final de los resultados de las pruebas hechas en la fase anterior, pudiendo obtener unas conclusiones satisfactorias de esta evaluación.

7.3.2 Presupuestos

En este punto se va a explicar el coste total que tendría desarrollo y despliegue del proyecto que se está exponiendo, en dos escenarios distintos. El primero de ellos expondrá los costos que supondría realmente llevar a cabo el despliegue del proyecto con materiales nuevos. El segundo de los presupuestos será más acorde con la realidad, teniendo en cuenta que prácticamente la totalidad del hardware utilizado para llevar a cabo este despliegue, ha sido material reutilizado de los antiguos equipos de las aulas del Laboratorio del Departamento de Informática de la Universidad Carlos III de Madrid.

El presupuesto referente al personal requerido para el desarrollo y despliegue de este proyecto se tomará como constate en ambos tipos de presupuestos ya que se considera de probada experiencia el personal que lo ha llevado a cabo.

Gastos de personal imputables al proyecto

Para realizar el cálculo del presupuesto se han tenido en cuenta las siguientes consideraciones:

- Jornada laboral de 4 horas.
- 5 días laborables a la semana (excluidos fines de semana y festivos).
- Las vacaciones y días festivos considerados en la planificación son:

  - Del 23 de Diciembre al 7 de Enero: Navidad.
  - 20 y 21 de Marzo: jueves y viernes Santo.
  - 1 de Mayo: Día del trabajador.

- Periodo de realización:
  - Inicio del proyecto: 6 de Octubre de 2008.
  - Finalización del proyecto: 10 de Mayo de 2009
- El proyecto requiere un analista, un programador y un técnico.
- Coste de personal por hora de trabajo:
  - Analista: 50 €/hora.
  - Analista-programador: 30 €/hora.
  - Técnico: 18 €/hora.

Por lo tanto, el cómputo total de horas dedicadas al proyecto es 592 horas, por lo que el proyecto ha tenido una duración total de 148 días reales.

Actividades	Duración (horas)	Recursos	Total (Euros)
Fase 1 (Estudio previo)	75	Analista	3.750 €
Fase 2 (Despliegue de arquitectura)	90	Técnico	1.620 €
	110	Analista-Programador	3.300 €
Fase 3 (Instalación de software)	90	Analista-Programador	2.700 €
Fase 4 (Realización de pruebas)	45	Analista-Programador	1.350 €
Fase 5 (Evaluación de pruebas)	64	Analista	3.250 €
Documentación	118	Analista	5.850 €
Total	592 horas (148 días)		21.820 €

Tabla 6: Especificación de actividades y coste.

Costes del hardware para el desarrollo del sistema

El despliegue del cluster que ha sido llevado a cabo ha requerido de los siguientes materiales y hardware:

- Equipo servidor:
  - Pentium III, Dual de 1 Ghz de velocidad y 2 GBytes de memoria.
- 34 nodos:
  - AMD K7, 1,7 Ghz de velocidad y 1 GBytes de memoria.
- Un switch de 24 bocas de 100 megabits por segundo.
- Un switch de 24 bocas de 1gigabit por segundo
- 16 tarjetas de red de velocidad gigabit.

- Periféricos:
    - 3 monitores con tecnología TFT.
    - 3 teclados.
    - 1 ratón óptico.

A continuación se expone una tabla de los costes del material tanto en el mercado, como el coste real. Cabe destacar que para la realización de este proyecto se han reutilizado múltiples recursos del Laboratorio del Departamento de Informática, en su mayoría no utilizados, lo que ha provocado una reducción drástica del gasto de este proyecto.

Concepto	Presupuesto 1 (Coste en el mercado)	Presupuesto 2 (Coste real)
Servidor	400 €	0 €
Nodo (34)	200 € (6.800 €)	0 €
Switch 100 Mbps	150 €	0 €
Switch gigabit	230 €	230 €
Tarjeta de red gigabit (16)	13 € (208 €)	13 € (208 €)
Monitor TFT (3)	105 € (315 €)	0 €
Teclado (3)	5 € (15 €)	0 €
Ratón óptico	15 €	0 €
Total	**8.133 €**	**438 €**

Tabla 7: Costes del material

Por tanto el ahorro conseguido es considerable, ya que mediante la reutilización de equipos y materiales que habían quedado prácticamente obsoletos para su uso docente se ha reducido el coste en 7.695 euros (siete mil seiscientos noventa y cinco euros).

Coste total del proyecto

La siguiente tabla detalla el coste total del proyecto:

Concepto	Presupuesto 1 (€)	Presupuesto 2 (€)
Recursos humanos	21.820 €	21.820 €
Material	8.133 €	438 €
Total antes de riesgo	**29.953 €**	**22.258 €**
Riesgo (10%)	2.995,30 €	2.225,80 €
Total antes de beneficio	**32.948,30 €**	**24.483,80 €**
Beneficio (15%)	4.942,25	3.672,57
Total final	**37.890,55 €**	**28.156,37 €**

Tabla 8: Coste total del proyecto.

El presupuesto número 1, contando con el material comprado y sin reutilización se estima en treinta y siete mil ochocientos noventa euros, con cincuenta y cinco céntimos.

El presupuesto número 2, habiendo reutilizado gran parte del material requerido para el despliegue del proyecto, asciende a veintiocho mil ciento cincuenta y seis euros, con treinta y siete céntimos.

APÉNDICE I

En este punto se expone la organización que va a tener la entrega, y se especificará el contenido de cada una de las carpetas que van a aparecer.

El índice de carpetas que se ha incluido en la entrega tiene el siguiente formato que se puede observar en la Figura 44.

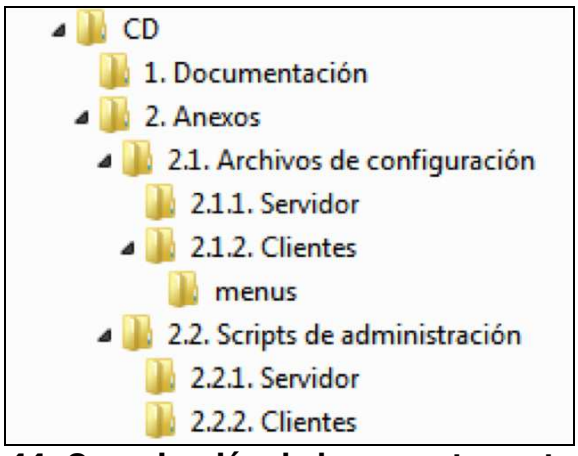

Figura 44: Organización de las carpetas entregadas

A continuación se expone resumido el objetivo y el contenido de cada una de las carpetas que aparecen en la figura anterior.

CD: Carpeta que contiene las dos principales carpetas que forman la documentación de este proyecto.

1. **Documentación**: En esta carpeta se puede encontrar el presente documento, en formato PDF.

2. **Anexos**: En este directorio están contenidas las carpetas que contienen tanto los archivos de configuración, como los scripts de administración generados durante el proyecto.

   2.1. **Archivos de configuración**: En esta carpeta se pueden encontrar los principales archivos de configuración de las aplicaciones instaladas en el cluster.

      2.1.1. **Servidor**: En esta carpeta están contenidos los archivos de configuración de las principales aplicaciones del servidor.

      2.1.2. **Cliente**: En esta carpeta están contenidos los archivos de configuración de las principales aplicaciones de los nodos.

         2.1.2.1. **Menus**: Se ha creado un script de arranque de kernel y ha sido necesario generar esta carpeta desde la que se copian ficheros de inicio como el “fstab” y el “menu.lst” del *grub*.

   2.2. **Scripts de administración**: En esta carpeta se pueden encontrar los scripts generados para facilitar la gestión y administración del cluster.

      2.2.1. **Servidor**: En esta carpeta se encuentran los scripts de administración del servidor del cluster.

      2.2.2. **Cliente**: En esta carpeta se encuentran los scripts de gestión y arranque de aplicaciones generados para los nodos del cluster.

REFERENCIAS

A continuación se exponen los recursos y la documentación que se han consultado y estudiado para el análisis y desarrollo del proyecto. Estas referencias aparecen a lo largo de todo el documento con un número entre corchetes.

[1] Ley de Amdahl: http://home.wlu.edu/~whaleyt/classes/parallel/topics/amdahl.html
[2] Proyecto ARPANET: http://www.iso.port.ac.uk/~mike/docs/internet/internet/node4.html
[3] Protocolo TCP/IP: http://www.yale.edu/pclt/COMM/TCPIP.HTM
[4] BSD (Berkeley) Unix:
http://www.freebsd.org/doc/es_ES.ISO8859-1/articles/explaining-bsd/article.html
[5] ARCNET: http://www.arcnet.com/abtarc.htm
[6] Kronenberg, N. P., Levy, H. M., and Strecker, W. D. 1985. VAXclusters (extended abstract): a closely-coupled distributed system. SIGOPS Oper. Syst. Rev. 19, 5 (Dec. 1985), 1. DOI= http://doi.acm.org/10.1145/323627.323628
[7] Tandem Himalaya:
http://www.redbarnhpc.com/v2/index.php/cluster-computing/the-history-of-cluster-computing
[8] IBM S/390 Parallel Sysplex: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=508100
[9] Parallel Virtual Machine (PVM): http://www.csm.ornl.gov/pvm/
[10] Arquitectura Beowulf: http://www.beowulf.org/overview/index.html
[11] Wang, D., Zheng, W., and Xiong, J. 1997. Research on cluster of workstations. In Proceedings of the 1997 international Symposium on Parallel Architectures, Algorithms and Networks (December 18 - 20, 1997). ISPAN. IEEE Computer Society, Washington, DC, 275.
[12] Linux Cluster Manager (LCM): http://linuxcm.sourceforge.net/
[13] Protocolo SSH: http://www.ssh.com/
[14] Protocolo RSH: http://www.protocolbase.net/protocols/protocol_RSH.php
[15] Protocolo RLOGIN:

http://www.lincoln.edu/math/rmyrick/ComputerNetworks/InetReference/111.htm
[16] Network Information Service (NIS):
http://www.freebsd.org/doc/es_ES.ISO8859-1/books/handbook/network-nis.html
[17] Domain Name System (DNS): http://www.dns.net/dnsrd/
[18] Local Area Network (LAN): http://kb.iu.edu/data/aesx.html
[19] Torque: http://www.clusterresources.com/products/torque-resource-manager.php
[20] Condor: http://www.cs.wisc.edu/condor/
[21] Portable Batch System (PBS): http://hpc.sissa.it/pbs/pbs.html
[22] OpenPBS: http://www.openpbs.org/
[23] Message Passing Interface (MPI): http://www.mcs.anl.gov/research/projects/mpi/
Corbett, P., Feitelson, D., Hsu, Y., Prost, J.P., Snir, M., Fineberg, S., Nitzberg, B., Traversat, B., Wong, P.: *MPI-IO : A parallel file I/O interface for MPI.* Technical Report NAS-95-002, NASA Ames Research Center, Moffet Field, CA (January 1995)
[24] MPICH: http://www.mcs.anl.gov/research/projects/mpi/mpich1/
[25] Open Multi-Processing (OpenMP): https://computing.llnl.gov/tutorials/openMP/
[26] Xen: http://www.xen.org/
[27] Paravirtualización: http://www.xtech.com.ar/articulos/xen/articulo-xen-coletti-html/x18.html
[28] Intel-VT: http://www.intel.com/technology/virtualization/
[29] AMD-V: http://www.amd.com/es-es/Processors/ProductInformation/0,,30_118_8796_14287,00.html
[30] Network File System Protocol (NFS): http://nfs.sourceforge.net/nfs-howto/
[31] Remote Procedure Call Protocol (RPC): http://www.faqs.org/rfcs/rfc1050.html
[32] Oracle Cluster File System (OCFS2): http://oss.oracle.com/projects/ocfs2/
[33] Real Application Cluster (RAC): http://www.oracle.com/lang/es/database/rac_home.html
[34] Internet Small Computer System Interface (ISCSI): http://www.ietf.org/rfc/rfc3720.txt
[35] Internet Engineering Task Force (IETF): http://www.ietf.org/
[36] Lustre File System: http://www.sun.com/software/products/lustre/
[37] Portable Operating System Interface (POSIX): http://standards.ieee.org/regauth/posix/
[38] Parallel Virtual File System (PVFS): http://www.pvfs.org/
[39] Benchmarking: Wong, P., der Wijngaart, R.: *NAS Parallel Benchmarks I/O Version 2.4.* Technical Report NAS-03-002, NASA Ames Research Center, Moffet Field, CA (January 2003).
http://www.nas.nasa.gov/News/Techreports/2003/PDF/nas-03-002.pdf
[40] NAS PARALLEL BENCHMARK: http://www.nas.nasa.gov/Resources/Software/npb.html
[41] FLASH I/O Benchmark Routine: http://flash.uchicago.edu/~zingale/flash_benchmark_io/
[42] COLLPERF:
[43] ROMIO: http://www.mcs.anl.gov/research/projects/romio/
[44] Linpack HPL: http://www.netlib.org/benchmark/hpl/
[45] Basic Linear Algebra Subprograms (BLAS): http://www.netlib.org/blas/