
***Facultad de Ciencias***

# Sistema de aprovisionamiento automático de servidores físicos para entorno de computación y almacenamiento de altas prestaciones

## (Automatic provisioning system for physical servers for high-performance computing and storage environments)

**Trabajo de Fin de Grado**
**para acceder al**

**GRADO EN INGENIERÍA INFORMÁTICA**

**Autor: Daniel Padilla Acebo**

**Director: José Ángel Herrero Velasco**

**Co-Director: Antonio Santiago Cofiño González**

**Septiembre - 2020**

# Agradecimientos

Tras la realización de este TFG es necesario expresar mis agradecimientos a todas las personas que me han ayudado directa o indirectamente a lo largo del grado.

En primer lugar, como no puede ser de otra manera, me gustaría agradecerles a mis padres, mis abuelos y a mi hermano, por haber hecho siempre todo lo posible para facilitarme el camino estos últimos años.

En segundo lugar, gracias a Fernando, compañero del grupo de Meteorología de la Universidad de Cantabria con el que he trabajado día a día durante todo este Trabajo de Fin de Grado, gracias por enseñarme y ayudarme a lo largo de todo este tiempo.

En tercer lugar, al codirector de este proyecto, por abrirme las puertas del que ha sido y será mi entorno de trabajo durante los próximos años, gracias por ofrecerme la posibilidad de formar parte de un grupo de trabajo excepcional, además de, por supuesto, por su ayuda durante este último año.

En cuarto lugar, al director de este proyecto, por su atención y disponibilidad durante la realización de este TFG, y en especial, por su gran trabajo como profesor en este grado.

Por último, no me puedo olvidar de todos los compañeros con los que he compartido estos momentos de mi vida, de los que me guardo un gran recuerdo, especialmente, de aquellos con los que, estoy seguro, mantendré una fuerte amistad a lo largo de los años.

# Resumen

El aprovisionamiento de servidores físicos consiste en su puesta a punto para su integración en una infraestructura de servidores existente. Esta puesta a punto incluye la instalación del sistema operativo (SO), la configuración de su red y de servicios, todos ellos necesarios para su funcionamiento dentro de la infraestructura.

Este proyecto tiene por objetivo desplegar la configuración necesaria para integrar una nueva infraestructura de almacenamiento de altas prestaciones en un servicio de gestión de datos climáticos ubicado en un centro de procesamiento de datos existente y, además, desarrollar una solución software que nos permita gestionar y optimizar el ciclo de vida de la infraestructura de servidores de forma automática, reproducible y fiable.

El objetivo último es llevar a cabo un despliegue eficiente y desatendido del SO y de los servicios, de manera consistente y segura en toda la infraestructura y con una intervención manual mínima.

**Palabras clave:** Centro de procesamiento de datos, aprovisionamiento, Ansible, PXE, Lustre

# Abstract

Provisioning physical servers consists of setting them up for integration into an existing server infrastructure. This process includes the installation of the operating system (OS) and the configuration of its network and services, all of which are necessary for its operation within the infrastructure.

The objective of this project is to deploy the necessary configuration to integrate a new high-performance storage infrastructure in an existing data processing center, and also to develop a software solution that allows us to manage and optimize the infrastructure life cycle.

The goal is to perform an efficient and unattended deployment of the OS and services, consistently across the entire infrastructure and with minimal manual intervention.

**Keywords**: Data Center, Provisioning, Ansible, PXE, Lustre

# Índice general

Índice de ilustraciones ..... 1
1. Introducción ..... 2
1.1 Antecedentes y motivación ..... 2
1.2 Objetivos ..... 3
1.3 Estructura del Documento ..... 4
2. Conceptos y tecnología ..... 5
2.1 Computación de altas prestaciones ..... 5
2.2 Computación de alta disponibilidad ..... 5
2.3 Aprovisionamiento ..... 7
2.3.1 Estado del arte ..... 7
2.3.2 Ansible ..... 8
2.3.2.1 Como funciona ..... 8
2.3.3 Repositorio ..... 9
3. Infraestructura ..... 10
3.1 Infraestructura de Cálculo ..... 10
3.2 Infraestructura de Almacenamiento ..... 11
3.3 Servicios del clúster ..... 12
3.3.1 Servicio de inicio de sesión único: SSO ..... 12
3.3.2 Servicio de directorio de trabajo de los usuarios: *Home Directory* ..... 14
3.3.3 Servicio de configuración de red: DHCP ..... 14
4. Componentes del despliegue ..... 16
4.1 Despliegue de servidores y servicios del clúster ..... 17
4.1.1 Servidor PXE ..... 17
Cómo funciona ..... 17
Nodo anfitrión ..... 19
4.1.1.1 Servidor DHCP ..... 19
4.1.1.2 Servidor TFTP ..... 20
4.1.1.3 Servidor FTP ..... 23
4.2 Despliegue del sistema operativo en los nodos de almacenamiento ..... 25
4.3 Despliegue de la configuración y servicios de los nodos de almacenamiento ..... 26
4.3.1 Sistema de ficheros ZFS ..... 26
Sistema de ficheros basado en pool de discos ..... 26
Snapshots ..... 27
RAID-Z ..... 27
Integridad de datos y reparación automática ..... 27
4.3.1.1 Diseño de la configuración ..... 28

4.3.1.2 Despliegue ..... 29
4.3.1.3 Integración en el aprovisionamiento automático ..... 30
4.3.2 Identificación y autentificación de usuarios ..... 31
4.3.2.1 Despliegue e Integración en el aprovisionamiento automático ..... 31
4.3.3 Directorio de trabajo remoto de los usuarios ..... 32
4.3.4 Sistema de ficheros Lustre ..... 33
4.3.4.1 ¿Qué es? ..... 33
4.3.4.2 Arquitectura Lustre ..... 33
4.3.4.3 Diseño de la arquitectura Lustre en el clúster ..... 35
Servidores de almacenamiento de objetos (OSS01 - OSS12) ..... 35
Recomendaciones para el almacenamiento OST ..... 35
Servidor de metadatos y Servidor de administración (MDS y MGS) ..... 36
Recomendaciones para el almacenamiento MDT y MGT ..... 37
4.3.4.4 Despliegue ..... 38
Instalación del software Lustre ..... 38
Creación del target de almacenamiento (MGT, MDT o OST) ..... 39
Creación del sistema de ficheros Lustre ..... 39
Inicio del servicio (MGS, MDS o OSS) ..... 39
4.3.4.5 Integración en el aprovisionamiento automático ..... 39
5. Conclusiones y líneas futuras ..... 41
6. Referencias ..... 42
Anexo A: Primer script de instalación del Sistema de ficheros raíz ZFS ..... 43
Anexo B: Segundo script de instalación del Sistema de ficheros raíz ZFS ..... 46

# Índice de ilustraciones

Ilustración 1: Representación de la arquitectura de un sistema de ficheros NFS. Fuente: developer.ibm.com ..... 5
Ilustración 2: Servidor Blade de 7U con capacidad para 20 nodos. Fuente: Supermicro ..... 11
Ilustración 3: Servidores destinados a albergar la nueva infraestructura de almacenamiento en el CPD. Fuente: Supermicro ..... 12
Ilustración 4: Protocolo DHCP. Fuente: Huawei ..... 15
Ilustración 5: Comunicación cliente-servidor en el protocolo PXE. Fuente: Intel PXE Specification Version 2.1 ..... 19
Ilustración 6: Fichero de configuración del servicio DHCP del nodo NAT. ..... 20
Ilustración 7: Archivo de configuración del servicio xinetd en el nodo NAT. ..... 22
Ilustración 8: Archivo grub.cfg empleado por el gestor de arranque GRUB del servidor TFTP en el nodo NAT. ..... 23
Ilustración 9: Menú de arranque de un nodo del CPD. ..... 26
Ilustración 10: Arquitectura del sistema de ficheros Lustre. Fuente: wiki.lustre.org ..... 35
Ilustración 11: Configuración de los servidores OSS del sistema de ficheros Lustre desplegado en el CPD. ..... 37
Ilustración 12: Configuración de los servidores MDS y MGS del sistema de ficheros Lustre desplegados en el CPD. ..... 38
Ilustración 13: comando usado para la creación del target MGT. ..... 39
Ilustración 14: comando usado para la creación del sistema de ficheros Lustre del servicio MGS. ..... 39
Ilustración 15: comando usado para iniciar el servicio MGS en el nodo. ..... 40

# 1. Introducción

Este Trabajo de Fin de Grado (TFG) se ha realizado dentro del Grupo de Meteorología y Computación del Departamento de Matemática Aplicada y Ciencias de la Computación de la Universidad de Cantabria.

Este grupo está formado por profesores e investigadores de la Universidad de Cantabria y el Consejo Superior de Investigaciones Científicas (CSIC, Instituto de Física de Cantabria IFCA) y realiza actividades de investigación teórica y aplicada en temas relacionados con la predicción numérica del tiempo, análisis del clima, minería de datos y computación de altas prestaciones.

Las actividades de investigación del grupo están respaldadas por una importante infraestructura de computación y almacenamiento y concentrada en el actual Servicio de Datos Climáticos, la cual tiene unas características que no difieren sustancialmente de las que pueden encontrarse en grandes clústeres de ordenadores de otros entornos de producción corporativos (procesadores, red InfiniBand, almacenamiento distribuido, etc.). Esta infraestructura está en pleno funcionamiento desde 2006 y está ubicada en el seno del CPD[1] 3Mares de la Facultad de Ciencias, Universidad de Cantabria.

De forma resumida, actualmente esta infraestructura tiene una capacidad de cómputo de unos 402 núcleos y 402 TB de almacenamiento distribuido y compartido.

## 1.1 Antecedentes y motivación

Como ya se ha mencionado, esta infraestructura, gestionada por el Grupo de Meteorología y Computación, lleva más de una década en funcionamiento. Durante este tiempo ha evolucionado de forma incremental y continua tanto en el ámbito del hardware como del software, para mantener activos todos los servicios ofrecidos por el Servicio de Datos Climáticos.

Esta evolución ha consistido principalmente en tareas de mantenimiento, como la sustitución de hardware defectuoso (discos, memoria, procesadores, etc ...) y la actualización de elementos software (sistema operativo, aplicaciones, servicios, ...), así como también la explotación de la capacidad de expansión de la infraestructura original.

A pesar de ello, ciertos servicios de la infraestructura, aunque siguen activos, progresivamente se han ido quedando insuficientes u obsoletos, en términos de capacidad y/o rendimiento, acordes con el incremento de las necesidades de los usuarios de los servicios del clúster.

[1] Centro de Procesamiento de Datos

Esto es lo que ha ocurrido con uno de sus servicios más críticos, como es la infraestructura de almacenamiento compartido, que fue desplegada en el año 2010. Esta infraestructura consiste en un sistema de ficheros distribuido[2] y compartido mediante el protocolo NFS[3]. Esta arquitectura cliente-servidor permite a los clientes acceder a archivos almacenados en un servidor remoto exactamente de la misma manera que un usuario accedería a cualquier archivo local.

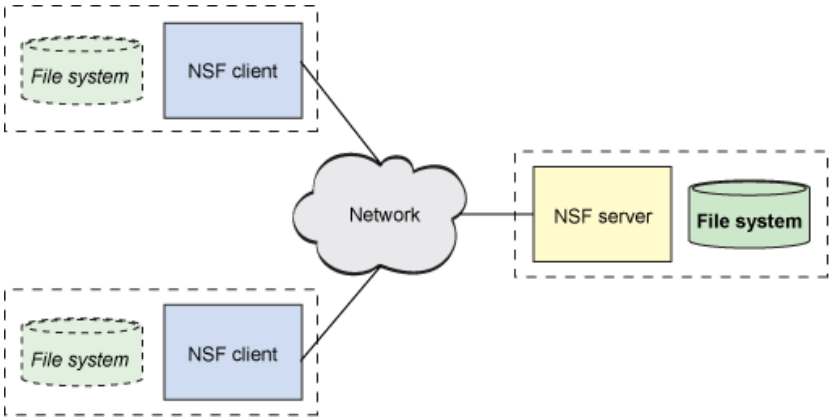


Ilustración 1: Representación de la arquitectura de un sistema de ficheros NFS. Fuente: developer.ibm.com

Como se observa en la ilustración 1, un servidor, el cual alberga físicamente los datos a compartir, exporta un sistema de ficheros local para que sea accesible a los clientes a través de la red TCP/IP. Esta arquitectura de almacenamiento compartido, presente en la actual infraestructura de almacenamiento del grupo, tiene como principal limitación la capacidad de escalar su rendimiento con respecto al número de clientes. Es decir, según aumenta el número de clientes simultáneos, aumenta la demanda de tráfico y rendimiento sobre un único servidor y almacenamiento físico, lo cual supone un importante cuello de botella en sus prestaciones.

Por este motivo, entre otros, se inició en su día un nuevo proyecto para dotar de una nueva infraestructura de almacenamiento que evitase las limitaciones de la infraestructura de almacenamiento NFS original.

## 1.2 Objetivos

Como se ha mencionado en el apartado anterior, el primer y principal objetivo es desplegar una nueva infraestructura de almacenamiento en el clúster existente, que cumpla con las demandas de capacidad y rendimiento de los usuarios de dicha infraestructura, además de proporcionar una mayor escalabilidad que garantice su usabilidad y gestión en el futuro a medio y largo plazo.

Como segundo objetivo, se plantea la creación de una plataforma o infraestructura software de aprovisionamiento que permita automatizar todas las operaciones necesarias para la

[2] https://es.wikipedia.org/wiki/Sistema_de_archivos_distribuido
[3] https://en.wikipedia.org/wiki/Network_File_System

instalación y configuración (despliegue) de nuevas máquinas y que sea capaz, de forma automática, de alcanzar un estado de funcionamiento del sistema deseado.

## 1.3 Estructura del Documento

El objeto de esta memoria es detallar el desarrollo y los resultados obtenidos durante el TFG. Para ello la memoria se estructura en los siguientes capítulos:

1. **Introducción**: disposición del entorno de trabajo, las motivaciones y los objetivos del proyecto.

2. **Conceptos y tecnología**: explicación necesaria para comprender el entorno y el desarrollo del proyecto.

3. **Infraestructura**: expone de forma detallada la infraestructura presente en el Servicio de Datos Climáticos previo al desarrollo del proyecto. Esta parte nos sirve como ejemplo de los diferentes componentes que conforman un clúster de computadores. Además, comprender la función de cada uno de estos componentes es necesario para justificar y desarrollar el proceso de aprovisionamiento que se expone en el siguiente capítulo.

4. **Componentes del despliegue**: desarrolla en detalle el proceso de diseño, despliegue e integración en el sistema de aprovisionamiento automático de cada uno de los componentes necesarios para alcanzar el estado deseado de la nueva infraestructura de almacenamiento.

5. **Conclusiones y líneas futuras**: recoge el estado tras la finalización del TFG y desarrolla una serie de conclusiones y líneas de trabajo en el futuro.

6. **Referencias**: bibliografía consultada y referenciada en este documento.

# 2. Conceptos y tecnología

El objetivo de este capítulo es realizar una breve introducción a algunos de los principales componentes y tecnologías empleados en el entorno de un CPD de las características de 3Mares, cuyas infraestructuras computacionales están especializadas en entornos HPC.

## 2.1 Computación de altas prestaciones

La computación de altas prestaciones [1][2] (HPC, de sus siglas en inglés *high performance computing*) consiste en el uso de múltiples recursos computacionales interconectados entre sí, que trabajan de forma paralela para la resolución de un objetivo común de la manera más eficiente y rápida posible.

Un supercomputador o clúster HPC está compuesto por un conjunto de computadores más pequeños:

- Cada computador diferente se denomina nodo. Los nodos pueden tener diferentes roles dentro del clúster (almacenamiento, cálculo, interfaz de usuario, ...).
- Cada nodo tiene sus propios recursos locales (procesadores, memoria, almacenamiento, ...). Estos recursos pueden ser homogéneos, según el rol de los nodos.
- Los nodos pueden intercambiar datos entre sí a través de una red de comunicaciones de altas prestaciones, con un gran ancho de banda y baja latencia (i.e. InfiniBand).
- Disponen de al menos un sistema de gestión de recursos computacionales (sistema de gestión de trabajos por colas), sistemas de ficheros compartidos y monitorización centralizado fundamentalmente

Hoy en día, el HPC juega un papel muy importante en el campo de la ingeniería y ciencia informática, utilizándose para una amplia gama de tareas computacionalmente intensivas en diversos campos científicos y tecnológicos, como son la predicción meteorológica, evolución del clima y la aplicación de la ciencia de datos[4], entre otras muchas.

## 2.2 Computación de alta disponibilidad

En informática, el término disponibilidad se utiliza para describir el período de tiempo durante el cual un servicio está disponible, así como el tiempo que requiere un sistema para responder a una solicitud realizada por un usuario. La alta disponibilidad es una calidad del sistema u otro componente que asegura un alto nivel de respuesta y rendimiento operativo durante un período de tiempo determinado [3][4].

En la mayoría de ámbitos de uso, proporcionar un servicio ininterrumpido es un aspecto igual

[4] https://en.wikipedia.org/wiki/Climatology

de crítico o incluso más que el rendimiento del propio sistema. Es por esto por lo que en el campo de la supercomputación cada vez se asigna un mayor número de recursos para mejorar la disponibilidad del sistema frente a los diferentes fallos que pueden producirse.

Existen un gran número de escenarios que pueden causar la interrupción parcial o total del servicio para los usuarios de un CPD. En función de su origen se pueden clasificar en dos categorías:

1. **Fallos externos**: Producidos por la interrupción de un servicio proporcionado por una entidad ajena al propietario del CPD y fuera de su control, como son el suministro eléctrico o el servicio de acceso a internet. Se puede garantizar o mejorar la disponibilidad del servicio frente a estos fallos con medidas como el uso de UPS o generadores eléctricos de emergencia, la instalación de una segunda línea eléctrica redundante suministrada por otro proveedor, etc .

2. **Fallos internos**: Malfuncionamiento de un elemento interno al CPD, Se pueden distinguir de dos tipos:

    a. **Fallos en la infraestructura de soporte vital del CPD**. Producidos por problemas en los sistemas de soporte vital del CPD, como los sistemas de distribución eléctrica o de refrigeración.

    b. **Fallos en la infraestructura computacional del CPD**, producidos habitualmente en alguno de los componentes de la infraestructura computacional que alberga. Este tipo de fallos es el más habitual, y en función de la causa se pueden distinguir dos tipos:

        i. **hardware**: fallo de alguno de los componentes físicos de un nodo (procesadores, memoria, almacenamiento, ...).
        ii. **software**: fallo causado por el sistema operativo o alguna aplicación que se ejecuta en los nodos.

En el caso de los fallos internos existen una serie de tecnologías y arquitecturas diseñadas para garantizar la disponibilidad del servicio. Una de las tecnologías más usadas en entornos HPC, concretamente en la gestión de los sistemas de almacenamiento, es la conocida como "Redundant Array of Inexpensive Disks" (RAID)[5]. Esta técnica de almacenamiento combina varios dispositivos físicos de almacenamiento en una o más unidades lógicas con el objetivo de proporcionar redundancia de los datos. Esta tecnología, puede tolerar el fallo de uno o varios dispositivos de almacenamiento sin resultar en la pérdida de datos o interrupción del servicio.

Otra técnica para proporcionar alta disponibilidad es la configuración de conmutación por error o *failover*, como son las plataformas Linux HA o Ubuntu HA. Esta tecnología requiere la existencia de uno o varios nodos adicionales, además de una red de interconexión dedicada, para que en el caso de que uno o más de los nodos del clúster fallen, otros nodos puedan asumir el servicio de forma automática y en el mínimo intervalo de tiempo posible.

[5] https://en.wikipedia.org/wiki/RAID

Estas son dos de las tecnologías y principios más usados en cualquier clúster de computadores y que junto con otras técnicas están presentes de forma paralela en el entorno de un data center para garantizar la disponibilidad ante cualquier tipo de fallo.

## 2.3 Aprovisionamiento

El aprovisionamiento de un servidor es el proceso de configuración de un nodo para ser integrado en una infraestructura y poder desempeñar su función dentro de ella. El aprovisionamiento puede abarcar todas las operaciones necesarias hasta alcanzar el estado de funcionamiento final deseado.

El aprovisionamiento del servidor incluye la instalación del hardware físico en los racks disponibles del CPD, la instalación y configuración del software, incluidos el sistema operativo y las aplicaciones, y su conexión a middleware, redes y almacenamiento.

### 2.3.1 Estado del arte

Históricamente, el aprovisionamiento de la infraestructura se ha manejado normalmente de forma manual. Hoy en día las tecnologías de virtualización[6] y de contenedores[7] han mejorado sustancialmente el proceso de aprovisionamiento, al tiempo que han reducido la necesidad de un aprovisionamiento y administración de hardware y software frecuentes.

Sin embargo, estas tecnologías de infraestructura virtual no eliminan por completo el problema del aprovisionamiento, ya que en una primera instancia es la propia infraestructura que proporciona el entorno de virtualización la que debe ser configurada de forma manual. Por este motivo se crean soluciones de *Infrastructure as Code* (IaC, infraestructura como código) que ofrecen una solución, y permiten automatizar el proceso de aprovisionamiento [5].

IaC es la gestión y el aprovisionamiento de infraestructura a través de código, en lugar de usar procesos manuales. Con IaC, se crean archivos de configuración que contienen las especificaciones de la nueva infraestructura y es un software específico el encargado de realizar las tareas necesarias para completar la configuración definida.

Además de automatizar el proceso de aprovisionamiento, lo que facilita el trabajo del administrador, IaC presenta otra serie de ventajas, como garantizar el mismo estado final en todos los servidores. Además permite dividir su infraestructura en componentes modulares que pueden ser más tarde combinadas de diferentes maneras, lo que proporciona un mayor potencial a estas herramientas.

Automatizar el aprovisionamiento de cualquier infraestructura es el primer paso para

[6] https://en.wikipedia.org/wiki/Virtualization
[7] https://en.wikipedia.org/wiki/OS-level_virtualization

automatizar el ciclo de vida operativo de sus aplicaciones.

Con esta filosofía en mente, se desarrollan las herramientas (IaC) de administración de la configuración, para abordar la necesidad de implementar nuevos servidores con configuraciones y actualizaciones definidas, que permitan un proceso de automatización más fluido y manejable. Las herramientas IaC más extendidas hoy en día son Puppet[8], SaltStack[9], Chef[10] y Ansible.

### 2.3.2 Ansible

Ansible es una de las herramientas más modernas y populares utilizadas como solución de gestión de la configuración (IaC). Presenta una serie de características que justifican su elección por encima del resto de alternativas mencionadas [6]:

- **Sintaxis sencilla**: los scripts de administración de configuración de Ansible se denominan *playbooks*. La sintaxis del *playbook* se basa en YAML[11], que es un lenguaje de serialización de datos diseñado específicamente para que los seres humanos puedan leer y escribir fácilmente.

- **Requisitos mínimos**: para administrar un servidor con Ansible, este debe tener disponible únicamente el servicio SSH activo y Python 2.5 (o posterior) instalado. No hay necesidad de preinstalar ningún otro software en el host. La máquina de control (la que usa para controlar máquinas remotas) debe tener Python 2.6 (o posterior) instalado.
  Esto facilita enormemente el aprovisionamiento ya que los requisitos software (SSH y Python) de los nodos se cumplen tras la instalación de prácticamente cualquier sistema operativo Linux.

- **Escala hacia abajo**: Ansible se puede utilizar para administrar cientos o incluso miles de nodos. Sin embargo, usar Ansible para configurar un solo nodo es igual de sencillo, simplemente se escribe un *playbook* y se ejecuta en el nodo o nodos deseados.

- **Módulos incorporados**: Se puede usar Ansible para ejecutar comandos de *shell* arbitrarios en sus servidores remotos, sin embargo, el poder real de Ansible proviene de la colección de módulos de los que dispone. Existen un gran número de módulos para realizar tareas como por ejemplo instalar un paquete, reiniciar un servicio o copiar un archivo de configuración.

[8] https://en.wikipedia.org/wiki/Puppet_(company)
[9] https://en.wikipedia.org/wiki/Salt_(software)
[10] https://en.wikipedia.org/wiki/Chef_(software)
[11] https://en.wikipedia.org/wiki/YAML

#### 2.3.2.1 Como funciona

La unidad básica de ejecución se llama *playbook*, la cual recoge un conjunto de tareas que se han de ejecutar en orden en el host remoto. Cuando ejecutamos un *playbook* en uno o varios nodos siempre se sigue el mismo proceso de ejecución:

1. La máquina de control o nodo maestro abre conexiones SSH en paralelo a todos los nodos remotos.
2. Genera un script de Python que realiza la primera acción definida en el *playbook*.
3. Copia el script en los nodos remotos a través de la conexión SSH de forma paralela.
4. Ejecuta el script en los nodos de destino de forma paralela.
5. Espera a que se complete la ejecución del script en todos los nodos.

Una vez realizados estos 5 pasos se pasa a la siguiente tarea del *playbook* y repite los pasos 2-5, así para cada tarea hasta completar las tareas del *playbook*.

### 2.3.3 Repositorio

Todos los *playbooks*, scripts y contenidos adicionales que se han desarrollado durante la realización de este TFG para proporcionar el sistema de aprovisionamiento automático de los nodos, han sido almacenados y gestionados en un repositorio en GitLab[12]. Por motivos de privacidad, el acceso a los contenidos del repositorio está limitado a solo personal vinculada al proyecto.

En el capítulo 4, donde se expone el proceso de despliegue de los nodos y la integración de los componentes software en el aprovisionamiento automático, se hace referencia a algunos de los contenidos de este repositorio, añadiendo fragmentos de código si es necesario.

[12] https://gitlab.com/scds/server-provisioning

# 3. Infraestructura

Este capítulo describe la infraestructura informática sobre la cual se ha desarrollado este trabajo. Dicha infraestructura conforma el denominado Servicio de Datos Climáticos de la Universidad de Cantabria, y le permite a éste el análisis, procesado y almacenamiento de los datos.

Es importante comprender la infraestructura del clúster, para poder así realizar el despliegue del nuevo sistema de almacenamiento e integrarlo en la infraestructura actualmente en producción en el CPD 3Mares.

En producción desde 2006, el clúster presenta básicamente los componentes típicos de una infraestructura computacional de estas características (HPC), como son un conjunto de nodos de cálculo, nodos e infraestructura de almacenamiento y los servicios necesarios para el correcto funcionamiento del clúster.

## 3.1 Infraestructura de Cálculo

Dispone de 45 nodos modelo SuperMicro TwinBlade con doble socket y procesadores quad core, conectados entre sí mediante una red InfiniBand para optimizar las comunicaciones.

En una configuración de rack de servidor estándar, una unidad de rack o 1U define el tamaño mínimo posible de cualquier equipo. Es decir, 1U equivaldría a 1 nodo/computador. El principal beneficio y justificación del formato *blade*, utilizado en estos nodos de cálculo, se relaciona con la eliminación de esta restricción para reducir los requisitos de tamaño.

El formato *blade* permite compartir muchos componentes para minimizar el uso del espacio físico y la energía, sin dejar de tener todos los componentes funcionales necesarios para ser considerados como nodos completamente independientes.


Ilustración 2: Servidor Blade de 7U con capacidad para 20 nodos. Fuente: Supermicro

En la ilustración 2 se puede ver el formato *blade*. Concretamente se muestra un dispositivo

de tamaño 7U que puede albergar hasta 20 nodos, como los presentes en la infraestructura del clúster del Servicio de Datos Climáticos.

Esto permite lograr una mayor densidad de cómputo. En total, la infraestructura de cálculo cuenta con (45 x 8) 360 cores, concentrados en un espacio físico muy inferior a la capacidad de un rack de tamaño estándar (42U) como los que se encuentran en el CPD 3Mares.

## 3.2 Infraestructura de Almacenamiento

La infraestructura original descrita en el Capítulo 1 consiste en un nodo conectado a una cabina de discos SATA3, exportados a través de la red mediante el protocolo NFS. Esta arquitectura, con un único servidor, presenta limitaciones considerables en cuanto a escalabilidad, rendimiento y de disponibilidad, lo que motiva la integración de una nueva infraestructura de almacenamiento que ponga solución a dichos problemas.

El nuevo sistema de almacenamiento diseñado para sustituir al original presenta una arquitectura cliente-servidor completamente escalable en términos de capacidad y rendimiento, gracias a distribuir el almacenamiento físico, el tráfico de datos y la carga de procesamiento en un número de nodos proporcional a las características de la infraestructura. Es decir, la capacidad de almacenamiento del nuevo sistema, así como sus servicios, pasan a estar complemente distribuidos.

Esta nueva infraestructura está compuesta por 14 nodos separados en 2 grupos. Un grupo de 12 nodos, donde se distribuye el almacenamiento físico de los datos del sistema de ficheros para proporcionar altas prestaciones de almacenamiento. Y un segundo grupo, compuesto de dos nodos, donde se almacenan los metadatos del sistema de ficheros de forma redundante, para proporcionar alta disponibilidad.

Ilustración 3: Servidores destinados a albergar la nueva infraestructura de almacenamiento en el CPD. Fuente: Supermicro

En la ilustración 3 se observan dos modelos distintos; el de la izquierda corresponde con un equipo Supermicro twinserver SYS-6028TP-DNCTR[13] (2U, 2 nodos) similar al empleado para los servidores de metadatos en la nueva infraestructura de almacenamiento. El de la derecha,

[13] https://www.supermicro.com/en/products/system/2U/6028/SYS-6028TP-DNCTR.cfm

corresponde con un nodo Supermicro SSG-6048R-E1CR60L[14] (4U, 1 nodo) similar a los utilizados en el nuevo sistema de almacenamiento, caracterizado principalmente por la cabina de 60 discos SATA3 integrada.

Esta arquitectura se expone con mayor grado de detalle en el Apartado 4.5 sistema de ficheros lustre, donde se explica el proceso de diseño y despliegue de esta nueva infraestructura.

## 3.3 Servicios del clúster

Para hacer que un clúster de computadores sea completamente operativo y funcional, no basta solo con desplegar los nodos de cálculo y de almacenamiento. También es necesario desplegar una serie de servicios que proporcionan interfaces y protocolos para que los usuarios autorizados puedan hacer uso de una forma ordenada, segura y eficiente de todos los recursos del clúster.

En el clúster del Servicio de Datos Climáticos existen una serie de nodos destinados a albergar y gestionar estos servicios. Entre estos nodos, los diferentes servicios son distribuidos sobre una infraestructura virtual, donde cada servicio o grupo de servicios es albergado en un entorno virtual independiente. Sin entrar en detalle sobre la infraestructura de virtualización, se van a exponer a continuación los principales servicios desplegados en el clúster.

### 3.3.1 Servicio de inicio de sesión único: SSO

Single Sign-On (SSO) [7], es un sistema de autenticación centralizada que permite a los usuarios de un clúster iniciar sesión en cualquiera de sus nodos con unas credenciales únicas, almacenados y gestionadas por uno o varios servidores. Este servicio no solo se usa en el entorno de la computación de alto rendimiento, sino que suele estar presente en cualquier organización que ofrezca una red informática para sus usuarios.
Este sistema presenta una serie de ventajas en comparación con el sistema de autenticación local (las credenciales se almacenan en el disco duro local de la máquina) que se emplea habitualmente en los ordenadores de uso personal:

- Permite al usuario iniciar sesión con unas únicas credenciales (usuario y contraseña) en todos los nodos que estén dentro del dominio SSO.

- Permite a todos los nodos del dominio SSO identificar y validar a un usuario, lo que es muy útil en las comunicaciones entre nodos dentro de la red informática.

- En el entorno de la administración, agrupa toda la información de inicio de sesión de los usuarios en una o varias bases de datos centralizadas, lo que simplifica el tratamiento de esta información.

[14] https://www.supermicro.com/en/products/system/4U/6048/SSG-6048R-E1CR60L.cfm

Aunque existen muchas formas de configurar un entorno SSO, generalmente se requieren cuatro elementos fundamentalmente [8]:

- Un directorio centralizado, o almacén central de identidades, que contiene la identidad del usuario y la información de autorización. Las soluciones más comunes son los servicios de directorio basados en el Protocolo ligero de acceso a directorios (LDAP[15]). En entornos que combinan sistemas Windows, y Linux, la herramienta Microsoft Active Directory[16] es la opción más extendida.

- Una herramienta para administrar la información del usuario en el directorio, en el caso de Microsoft Active Directory, se usa el complemento MMC nativo de Windows "Usuarios y equipos de Active Directory" para administrar la información en el directorio.

- Un mecanismo para autenticar identidades de usuario. El protocolo LDAP puede autenticar usuarios directamente, pero es común usar el protocolo Kerberos. En entornos Windows, Active Directory proporciona acceso LDAP a las identidades de los usuarios y utiliza una versión personalizada de Kerberos[17] para su autenticación. En sistemas modernos UNIX y Linux, el proceso de validación de usuarios se lleva a cabo mediante la integración de diferentes piezas. Fundamentalmente, LDAP y kerberos como *backend* de gestión de credenciales y mecanismos de autenticación, NSS como mecanismo de identificación y el sistema Pluggable Authentication Module (PAM), como *interfaz* para los procesos de validación entre los usuarios y las aplicaciones. En sistemas Linux actuales, también se usa el servicio SSSD como una capa más de abstracción.

- Versiones de las rutinas de la librería de C que obtienen las credenciales de un usuario. Estas rutinas, en un entorno de inicio de sesión local leen archivos planos tales como /etc/passwd y /etc/group en sistemas operativos linux y devuelven respuestas basadas en su contenido. Las nuevas rutinas, en un entorno SSO, pueden leer fuentes de datos de diferentes orígenes como un directorio de datos centralizado.

Los elementos descritos se dividen en una infraestructura de servidor y de cliente. La infraestructura de servidor es la encargada de almacenar los datos de las identidades de los usuarios, ofrecer las herramientas para tratar dicha información y proporcionar el acceso a los datos. La infraestructura de cliente debe configurarse en todos aquellos nodos para los cuales se requiere el acceso a los datos almacenados en el directorio centralizado, lo que implica que formen parte del dominio SSO.

Para el inicio de sesión único (SSO), el clúster cuenta con dos nodos en configuración *failover*, con sistema operativo Windows, que utilizan la herramienta Microsoft Active Directory como directorio centralizado, el complemento MMC como herramienta de administración de la información y el protocolo Microsoft Kerberos para proporcionar la autenticación a los

[15] https://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protoco
[16] https://en.wikipedia.org/wiki/Active_Directory
[17] https://docs.microsoft.com/en-us/windows/win32/secauthn/microsoft-kerberos

usuarios.

El entorno descrito es relativamente complejo y emplea varias tecnologías para ofrecer un servicio SSO. Los objetivos de este TFG plantean únicamente la integración de los nuevos servidores en esta infraestructura de SSO, ya configurada y desplegada previamente. La configuración de los nuevos clientes Linux que forman parte del dominio SSO se describe en el apartado 4.3 identificación y autentificación.

### 3.3.2 Servicio de directorio de trabajo de los usuarios: *Home Directory*

Para que un usuario del clúster pueda trabajar en cualquiera de los nodos pertenecientes al dominio de inicio único de sesión (SSO), debe existir una infraestructura que proporcione al usuario un mismo directorio de trabajo, independientemente del nodo en el que inicie sesión.

En distribuciones Linux, este directorio es el llamado *home directory,* y sirve como depósito para los archivos, directorios y programas personales de un usuario. También es el directorio por defecto en el que un usuario se encuentra por defecto después de iniciar sesión en el sistema [7].

Para proporcionar este servicio, el clúster cuenta con una infraestructura de almacenamiento dedicada a albergar el contenido de estos directorios de trabajo de sus usuarios. Presenta una arquitectura similar a la infraestructura de almacenamiento compartido original (sistema de ficheros exportado a la red mediante el protocolo NFS), solo que ésta dispone de un tamaño inferior, acorde con las necesidades de almacenamiento de este servicio.

Esto supone que, a la hora de aprovisionar un nuevo nodo en el clúster, es necesario realizar el despliegue y configuración del software necesario para integrar el nodo en el servicio de directorio de trabajo remoto. Este proceso es parte del aprovisionamiento realizado en este TFG y se incluye detalladamente en el apartado 4.3.3.

### 3.3.3 Servicio de configuración de red: DHCP

Dynamic Host Configuration Protocol (DHCP) es un protocolo de administración de red utilizado en redes IP, mediante el cual un servidor, ejecutado en uno de los nodos de servicio, asigna dinámicamente una dirección IP y otros importantes parámetros de configuración de red a cada dispositivo o interfaz de red de un nodo cliente [7].

La principal ventaja de desplegar un servidor DHCP en un clúster es centralizar la gestión del espacio de direcciones del propio clúster en un solo nodo. De esta forma es mucho más sencillo realizar tareas de mantenimiento o modificaciones sobre la distribución del espacio de direcciones de las diferentes redes presentes al clúster o incluso, a todo el CPD.

En la práctica, esto significa que para realizar la configuración de red de los nuevos servidores de almacenamiento, basta con modificar el archivo de configuración del servidor DHCP y

añadir las direcciones de red estáticas asignadas a los nuevos nodos.

Otra ventaja es que permite formatear un nodo sin perder su configuración de red, recuperándose al reinstalar el sistema operativo. Esto es particularmente útil durante el proceso de integración de una nueva infraestructura en un clúster, cuando se realizan múltiples instalaciones hasta alcanzar el estado final deseado.

Un aspecto negativo a tener en cuenta sobre el protocolo y servicio DHCP es que un fallo en el servidor puede provocar la pérdida de acceso a la red de múltiples nodos. Por lo tanto, el servicio DHCP es una de las piezas críticas en el entorno del clúster.

DHCP es uno de los protocolos de red más sencillos. Su funcionalidad de divide en 4 pasos:

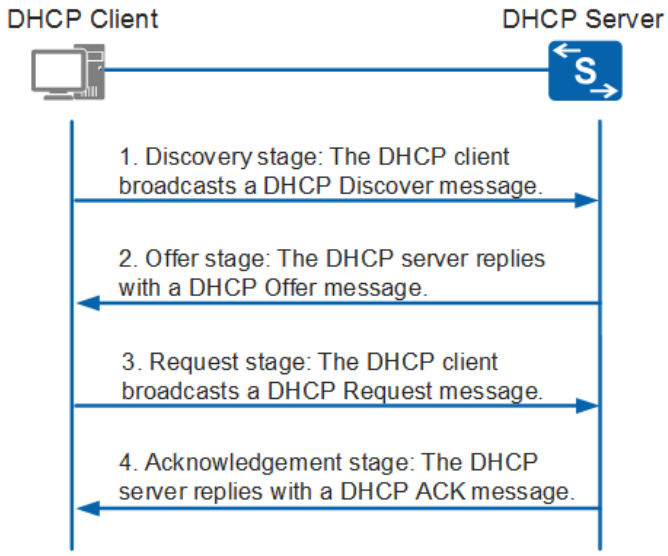


Ilustración 4: Protocolo DHCP. Fuente: Huawei

**Paso 1**. DHCPDISCOVER, el cliente transmite una solicitud *broadcast* a un servidor DHCP.

**Paso 2**. DHCPOFFER, los servidores DHCP de la red ofrecen una configuración de red al cliente.

**Paso 3**. DHCPREQUEST, el cliente transmite una solicitud para adquirir la configuración IP de la oferta.

**Paso 4**. DHCPACK, el servidor DHCP al que responde el cliente reconoce al cliente, le asigna las opciones DHCP configuradas y actualiza su base de datos DHCP. Luego, el cliente inicializa y vincula su pila de protocolos TCP/IP y puede comenzar la comunicación de red.

Como última característica a destacar sobre el protocolo DHCP, es que juega un papel fundamental en el aprovisionamiento automático de un servidor. Como se explica en el capítulo 4.1 Servidor PXE, DHCP es uno de los servicios necesarios para el despliegue de un servidor PXE, necesario para el aprovisionamiento automático.

# 4. Componentes del despliegue

En este capítulo son descritos los pasos necesarios para realizar el aprovisionamiento de un nodo en la infraestructura del clúster del Servicio de Datos Climáticos, así como las herramientas software desarrolladas (*playbooks* y scripts) para lograr el aprovisionamiento automático de cada uno de los diferentes servicios y herramientas software desplegados. Es importante, además, aportar una breve descripción de cada una de las tecnologías de las que se va a hablar a continuación; qué son, cómo funcionan y sobre todo qué funcionalidades ofrecen y cuál es su papel en la infraestructura del clúster.

El proceso de aprovisionamiento de cada uno de los nodos que forman la nueva infraestructura de almacenamiento en el clúster se puede dividir en dos etapas. La primera consiste en la instalación de un sistema operativo base en el nuevo nodo. Y la segunda, que comprende la instalación y configuración de todo el software necesario para lograr el estado final deseado en el nodo. Es decir, su integración completa en los servicios del clúster, para que éste pueda así desempeñar adecuadamente su papel dentro del clúster.

Para realizar la instalación de un sistema operativo de forma eficiente para múltiples nodos, es necesario desplegar en el clúster una serie de servicios que proporcionen la funcionalidad necesaria para instalar el sistema operativo desde una fuente remota y accesible para todos los nodos a través de la red de intercomunicaciones. Estos servicios forman un denominado servidor PXE, explicado con detalle en el apartado 4.1.1.

Para garantizar el correcto despliegue e integración en la infraestructura de aprovisionamiento automático de cada uno de los componentes del despliegue, se siguen los siguientes pasos:

- **Definición del objetivo**: establecer el estado final del elemento que se desea obtener en el nodo.

- **Métodos o herramientas a utilizar**: determinar la tecnología o herramientas software más adecuadas para lograr el estado final establecido.

- **Despliegue y pruebas**: desplegar de forma manual el software en el nodo de destino y comprobar que el estado es el deseado.

- **Integración en el aprovisionamiento automático**: desarrollar los scripts y/o *playbooks* necesarios para lograr el despliegue automático del software a través de la herramienta Ansible.

## 4.1 Despliegue de servidores y servicios del clúster

Para instalar un sistema operativo en un nuevo nodo es necesario que este tenga acceso a una fuente de instalación, el método tradicional consiste en insertar un dispositivo de almacenamiento externo que contenga la imagen de instalación en un puerto local del servidor (e.i. USB). Sin embargo, en el ámbito de la computación en clúster, se presenta la necesidad de instalar un sistema operativo en decenas o cientos de nodos, haciendo el método tradicional extremadamente ineficiente. Por esto surge la necesidad de desplegar un nuevo servicio en el clúster que proporcione a todos los nodos el acceso a una fuente de instalación a través de la red de interconexión, permitiendo realizar la instalación del sistema operativo de forma paralela en múltiples nodos y sin necesidad de tener acceso físicamente a los servidores del clúster en el CPD.

### 4.1.1 Servidor PXE

Preboot Execution Environment (PXE) es un estándar que define una comunicación cliente-servidor la cual permite a un equipo sin presencia de sistema operativo, descargar a través de la red una imagen de arranque mediante el protocolo Trivial File Transfer Protocol (TFTP). De esta forma, el cliente puede cargar la imagen de un sistema operativo o programa de instalación procedente de un medio de almacenamiento remoto, accesible desde dentro del clúster por cualquiera de sus nodos.

PXE emplea una serie de protocolos de internet muy estandarizados, como son UDP, DHCP y TFTP, que presentan una importante ventaja, y es la de ser fáciles de implementar en el firmware de la propia tarjeta de red. Es así como el protocolo PXE es capaz de establecer las comunicaciones de red cliente-servidor con los servidores DHCP y posteriormente TFTP para descargar y ejecutar una imagen de arranque sin necesidad de un sistema operativo [7][9].

Sin entrar en su funcionamiento, TFTP es un protocolo de transmisión de ficheros a través de la red que presenta como característica principal su especial simplicidad, necesaria para su integración en el protocolo PXE.

Gracias a su sencillez, este protocolo puede ser implementado en un número de líneas de código muy inferior en comparación con otros protocolos de funcionalidad similar (e.i. File Transfer Protocol (FTP)). Es importante recordar que esta característica es fundamental, debido a que todo el stack del protocolo PXE está implementado en el firmware de la tarjeta de red, donde el espacio de almacenamiento es un recurso extremadamente limitado.

#### Cómo funciona

La ilustración 5 muestra paso a paso el protocolo PXE [10]:

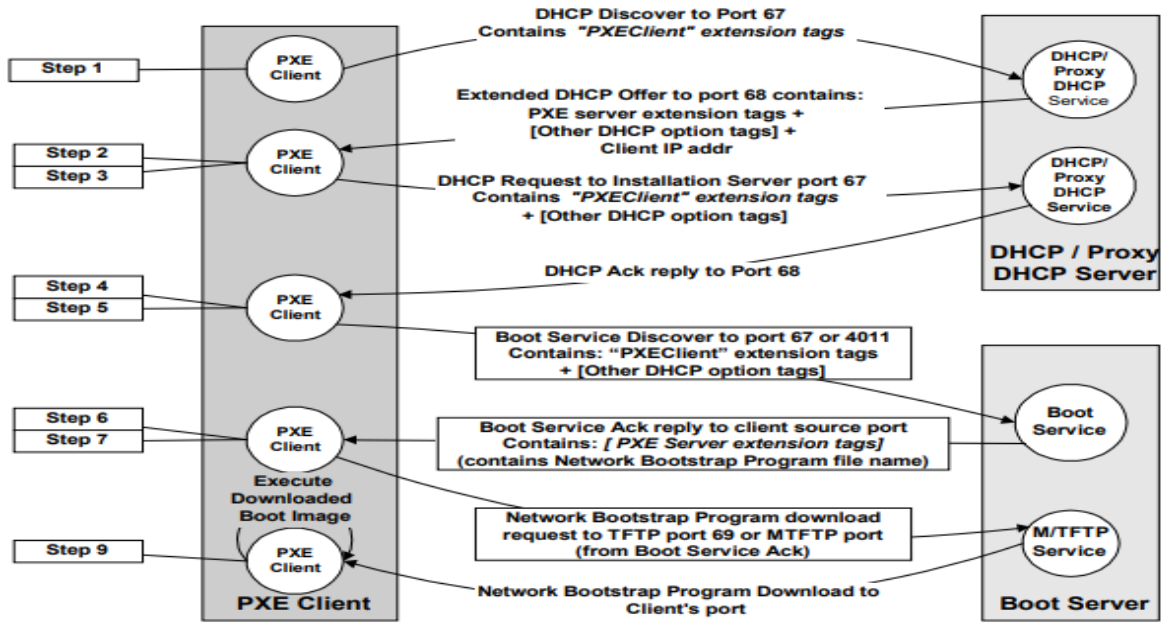


Ilustración 5: Comunicación cliente-servidor en el protocolo PXE. Fuente: Intel PXE Specification Version 2.1

Los primeros cuatro pasos corresponden con el protocolo DHCP (DHCPDISCOVER, DHCPOFFER, DHCPREQUEST y DHCPACK), a los que se añade un paso adicional, en el cual se proporciona de nueva información relativa al protocolo PXE, como la arquitectura del cliente o la dirección IP del servidor de arranque (servicio TFTP).

**Paso 1.** El cliente envía un paquete *broadcast* DHCPDISCOVER junto con varias extensiones adicionales al protocolo PXE como un identificador y el tipo de arquitectura.

**Paso 2.** El servidor responde con un mensaje DHCPOFFER con la configuración IP asignada para el cliente, además de varias extensiones PXE como la dirección IP del servidor de arranque.

**Paso 3.** El cliente guarda la información recibida en el paquete anterior y envía un paquete DHCPREQUEST para solicitar su configuración IP y confirmar la información recibida.

**Paso 4.** El servidor envía un paquete DHCPACK confirmando al cliente su configuración de red asignada.

Comienza la comunicación del cliente con el servidor de arranque:

**Paso 5.** El cliente envía un mensaje DHCPOFFER con la misma información que el paquete DHCPDISCOVER del paso 1.

**Paso 6.** El servidor de arranque devuelve un mensaje al cliente conteniendo con la información del nombre del fichero del gestor de arranque.

**Paso 7.** El cliente descarga el fichero indicado desde el servidor usando para ello el protocolo TFTP.

**Paso 8.** El cliente comprueba la autenticidad del fichero descargado.

**Paso 9.** Por último, el cliente ejecuta el fichero del gestor de arranque descargado.

Una vez el cliente ha ejecutado el gestor de arranque (i.e. GRUB), descarga los ficheros de kernel e initrd mediante el protocolo TFTP, a continuación, con el kernel ya cargado y ejecutándose en memoria, el cliente puede establecer una comunicación de tipo FTP o HTTP con el servidor donde se encuentra la imagen de instalación del sistema operativo que se desea instalar en el cliente.

### Nodo anfitrión

Como hemos visto, para proporcionar el servicio PXE descrito anteriormente, es necesario el despliegue de tres servicios diferentes (DHCP, TFTP y FTP o HTTP), todos ellos accesibles desde cualquiera de los nodos del clúster que quieran contar con esta funcionalidad.

Como se ha mencionado, ya existe un servicio DHCP desplegado en la infraestructura del clúster del Servicio de Datos Climáticos, albergado éste en el nodo NAT. En teoría, cada uno de estos servidores podría desplegarse en un nodo distinto, pero, por simplicidad se han desplegado los servidores TFTP y FTP también en el nodo NAT.

Una vez comprendido el funcionamiento básico del estándar y su uso e integración dentro del clúster, se expone con mayor detalle cada uno de los componentes necesarios para desplegar un servidor PXE que cumpla con las necesidades del CPD.

#### 4.1.1.1 Servidor DHCP

El servicio DHCP es uno de los principales requisitos en la infraestructura de cualquier clúster o data center. En nuestro caso, el clúster ya disponía de este servicio desplegado y configurado para suministrar las direcciones IP estáticas y dinámicas a los equipos dentro de las diferentes redes del clúster.

Por tanto, el trabajo realizado consiste en añadir a este servicio la configuración de funcionalidad necesaria para ejercer su papel dentro del protocolo PXE [11]:

```
class "pxeclients" {
        match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
        next-server 192.168.202.1;

        if option architecture-type = 00:07 {
                filename "grub/shim.efi";
        }
        else {
                filename "pxelinux.0";
        }
}
```

Ilustración 6: Fichero de configuración del servicio DHCP del nodo NAT.

En la ilustración 6, se muestran las líneas añadidas al fichero de configuración del servicio DHCP ya existente. Se distinguen los argumentos necesarios para la identificación del cliente PXE por parte del servicio DHCP y su correcta comunicación en función de las características del cliente.

- **option vendor-class-identifie**r: información sobre el cliente PXE, añadida en el mensaje DHCPDISCOVER y enviada en el Paso 1 del estándar para identificarse como un cliente PXE.

- **option architecture-type**: información sobre el cliente PXE, añadida en el mensaje DHCPDISCOVER y enviada en el Paso 1 del estándar para indicar el tipo de arquitectura del cliente PXE. En nuestro caso se emplea una estructura condicional para diferenciar clientes que implementan interfaz UEFI o BIOS.

Por otro lado, se pueden observar los atributos almacenados por el servicio DHCP sobre el servidor de arranque.

- **next-server**: dirección IP del servidor de arranque que se enviará al cliente en el Paso 2.

- **filename**: nombre del fichero correspondiente al gestor de arranque que se envía al cliente en el Paso 6 para que éste pueda descargarlo mediante el protocolo TFTP.

La configuración añadida permite al servicio DHCP detectar a un cliente PXE y proporcionarle la información necesaria sobre el servidor de arranque. El servidor DHCP descrito es suficiente para completar los pasos desde el 1 hasta el 6 del protocolo PXE.

#### 4.1.1.2 Servidor TFTP

La infraestructura actual del clúster no contaba con este servicio, por lo que ha sido necesario realizar todas las fases para diseñar y desplegar este nuevo servicio [11].

1. **Definición del objetivo**: desplegar un servidor de almacenamiento compartido dentro del clúster, accesible para todos los nodos mediante el protocolo TFTP. El nodo elegido para albergar este servicio es el nodo NAT.
   El servidor debe ofrecer los ficheros de arranque necesarios (gestor de arranque e imágenes de arranque), para que el cliente sea capaz de cargar y ejecutar un kernel.

2. **Métodos o herramientas a utilizar**: conociendo las características del nodo donde va a ser desplegado el servicio, debemos elegir una solución compatible, estable y segura para la familia y versión del Sistema Operativo presente en el equipo. En nuestro caso, el sistema operativo instalado en el nodo NAT es CentOS 5.

   En el caso de CentOS 5, además de los paquetes propios del servicio TFTP como son tftp y tftp-server también es necesario instalar y configurar el demonio xinetd, que será el encargado de escuchar las peticiones de red en el puerto asignado por defecto para

el protocolo TFTP y reenviarlas hacia el servicio encargado y asignado para resolver las peticiones.

3. **Despliegue y pruebas**: puesto que no vamos a modificar ningún servicio o característica críticos del clúster o del nodo, podemos efectuar el despliegue manual en el nodo NAT sin afectar a su funcionamiento.

   El despliegue consiste en la instalación de los paquetes indicados en la fase anterior y su configuración necesaria. En esta ocasión, la única configuración no por defecto que vamos a realizar es la correspondiente al servicio xinetd, modificando el parámetro que establece el directorio, dentro del nodo, donde se almacenarán los contenidos compartidos por el servidor TFTP.

```
service tftp
{
        socket_type             = dgram
        protocol                = udp
        wait                    = yes
        user                    = root
        server                  = /usr/sbin/in.tftpd
        server_args             = -s /var/lib/tftpboot -v -v -v -r blksize
        disable                 = no
}
```

Ilustración 7: Archivo de configuración del servicio xinetd en el nodo NAT.

La ilustración 7 corresponde con el fichero de configuración del demonio xinetd relativo al servidor TFTP. El argumento señalado es el directorio del nodo anfitrión (NAT) donde se almacenan los contenidos del servidor TFTP.

El último paso es añadir en el directorio designado todos los archivos necesarios para proporcionar la funcionalidad de arranque a través de red que estamos buscando. Como ya se ha explicado, estos contenidos son los archivos relativos al gestor de arranque y las imágenes de arranque, correspondientes al kernel (vmlinuz) y sistemas ramdisk (initrd) asociados.

**Gestor de arranque (bootloader)**: Como ya hemos mencionado, vamos a hacer uso del cargador de sistema GRUB. Sin entrar en excesivo grado de detalle sobre el proceso de arranque, son necesarios al menos dos archivos para completarlo:

- grubx64.efi: archivo binario ejecutable del gestor de arranque GRUB que podemos obtener de cualquier ISO de instalación de sistemas operativos Linux o descargando el paquete grub de los repositorios oficiales. Este fichero proporciona la funcionalidad necesaria para, entre otras cosas, cargar y comenzar la ejecución de las imágenes de arranque.

- grub.cfg: archivo de configuración del cargador de arranque GRUB, creado específicamente para cumplir con las necesidades del servidor PXE. Este fichero indica al gestor de arranque GRUB la localización de las imágenes de arranque (vmlinuz e initrd) que se desea cargar, además de una serie de

argumentos y opciones adicionales, como son la dirección de un servidor FTP, donde podrán encontrarse la imagen de instalación del sistema operativo y el fichero de kickstart[18]. Este último, suministra al programa de instalación una configuración determinada, lo que permite al cliente realizar el proceso de instalación de forma automatizada, sin necesidad de ninguna intervención externa.

```
menuentry 'centos7.7.1908 kickstart' {
        linuxefi kernels/centos7.7.1908/vmlinuz
 inst.ks=ftp://192.168.202.1/centos7.7.1908/kickstart/anaconda-ks_luno02.cfg
        initrdefi kernels/centos7.7.1908/initrd.img
}
```

Ilustración 8: Archivo grub.cfg empleado por el gestor de arranque GRUB del servidor TFTP en el nodo NAT.

Como se puede ver en la ilustración 8, se emplean las opciones linuxefi e initrdefi para indicar la localización de las imágenes de arranque vmlinuz e initrd.img presentes en el servidor TFTP.
La opción inst.ks indica la dirección de red del servidor FTP y el nombre de fichero del documento de kickstart, empleado durante la instalación del sistema operativo.

**Imágenes de arranque (boot images)**: es necesario proporcionar un kernel que permita llevar a cabo el proceso de arranque deseado.

- vmlinuz: kernel de linux comprimido capaz de ser cargado en memoria y comenzar el proceso de arranque del sistema operativo para la instalación.

- initrd.img: sistema de ficheros que se carga completamente en memoria RAM y funciona como el sistema de ficheros raíz durante los primeros pasos del proceso de arranque de un sistema operativo. Permite ejecutar programas y contiene una serie de módulos adicionales que el kernel puede cargar y utilizar. Una vez el sistema de ficheros raíz original ya pueda ser montado, este se sustituye por el initrd y comienza la fase final del proceso de arranque.

4. **Integración en el aprovisionamiento automático**: para automatizar el despliegue del servicio TFTP descrito, se ha desarrollado un *playbook* que utiliza una serie de módulos Ansible:

   - módulo yum[19]: se usa para instalar el software necesario (tftp-server, xinetd) para el servicio TFTP mediante el gestor de paquetes presente en el host de destino.

   - módulo template[20]: se usa para copiar el archivo de configuración del servicio xinetd (previamente almacenado en nodo maestro) en el host de destino.

[18] https://en.wikipedia.org/wiki/Kickstart_(Linux)
[19] https://docs.ansible.com/ansible/latest/modules/yum_module.html
[20] https://docs.ansible.com/ansible/latest/modules/template_module.html

- módulo synchronize[21]: se usa para sincronizar un directorio en el nodo maestro que contiene los contenidos del servidor TFTP (gestor e imágenes de arranque) con un directorio en el nodo remoto.

Si recordamos, el protocolo PXE define la comunicación entre un cliente y un servidor para conseguir que un equipo sin presencia de un sistema operativo, es decir, ejecutando únicamente el firmware presente en la tarjeta de red, sea capaz de descargar a través de la red una imagen de arranque y comenzar la ejecución de un sistema operativo.

Bajo esta definición, el servidor ya estaría completo al haber desplegado los servidores DHCP y TFTP. Sin embargo, en nuestro caso, cargar un sistema operativo remoto a través de la red no es el objetivo final, sino un requisito para poder desplegar y ejecutar un proceso de instalación que nos permita tener un sistema operativo de referencia en cualquiera de los nodos del clúster. Todo ello de forma paralela, desatendida y utilizando una misma fuente de instalación remota dentro del propio clúster.

Por lo tanto, para completar el servidor PXE será necesario desplegar un servicio más, un servidor FTP donde almacenar la imagen de Instalación del sistema o sistemas operativos que se desea que estén disponibles para el aprovisionamiento.

#### 4.1.1.3 Servidor FTP

Al igual que con el servidor TFTP, la infraestructura del clúster del Servicio de Datos Climáticos no cuenta con este servicio particular, por lo que será necesario llevar a cabo todas las fases para diseñar y desplegar este nuevo servicio en el clúster [11].

1. **Definición del objetivo**: desplegar un servidor de almacenamiento compartido dentro del clúster, accesible para todos los nodos mediante el protocolo FTP. El nodo elegido para albergar este servicio vuelve a ser el nodo NAT.
   El servidor debe ofrecer la imagen o imágenes de instalación de los sistemas operativos que se deseen.

2. **Métodos o herramientas a utilizar**: al igual que con el servidor TFTP, es necesario conocer las características del nodo donde va a ser desplegado el servicio y elegir una solución compatible, estable y segura para la familia y versión del Sistema Operativo presente en el equipo. En nuestro caso, el sistema operativo instalado en el nodo NAT es CentOS 5.

   Para el despliegue del servidor FTP utilizamos el paquete vsftpd, disponible en los repositorios estándar de CentOS 5. Para determinar la configuración del servicio, hay que tener en cuenta los requisitos funcionales, de rendimiento y de seguridad que debe presentar el servidor:

[21] https://docs.ansible.com/ansible/latest/modules/synchronize_module.html

- **requisitos funcionales**: acceso permitido de lectura a todos los archivos del servidor de forma anónima.

- **requisitos de rendimiento**: al tratarse de un servicio que únicamente va a ser usado para efectuar la instalación de un sistema operativo, no es una actividad recurrente y no es necesario prestar demasiada atención al rendimiento del mismo.

- **requisitos de seguridad**: permitir el acceso únicamente a clientes de dentro del clúster para evitar posibles brechas de seguridad.

3. **Despliegue y pruebas**: al igual que ocurría con el despliegue del servicio TFTP, no es necesario modificar ningún servicio o característica críticos del clúster o del nodo, por lo que podemos efectuar el despliegue manual en el nodo NAT sin afectar a su funcionamiento.

   Este despliegue consiste en la instalación del paquete vsftpd y su configuración necesaria. En esta ocasión, debemos modificar el archivo de configuración por defecto para garantizar los requisitos planteados en el paso anterior.

   Por último, es necesario añadir, en el directorio FTP, las fuentes de instalación que se requieran. En este caso se trata de la ISO de instalación de CentOS 7, versión *minimal*. Para hacerlo, es necesario descargar la ISO desde cualquiera de los repositorios oficiales disponibles y extraer todos los contenidos de la imagen respetando la estructura original dentro del directorio FTP designado.

   Para ofrecer la posibilidad de una instalación predefinida y desatendida, es necesario incluir también en el directorio FTP los archivos de kickstart que se deseen.

4. **Integración en el aprovisionamiento automático**: para automatizar el despliegue del servicio FTP descrito, se ha desarrollado un *playbook* que utiliza una serie de módulos Ansible:

   - módulo yum: se usa para instalar el software necesario (vsftpd) para el servicio FTP mediante el gestor de paquetes presente en el host de destino.

   - módulo template: se usa para copiar el archivo de configuración del servicio vsftpd (previamente almacenado en nodo maestro) en el host de destino.

   - módulo synchronize: se usa para sincronizar un directorio en el nodo maestro que contiene los contenidos del servidor FTP (imagen de instalación del sistema operativo y ficheros de kickstart) con un directorio en el nodo remoto.

## 4.2 Despliegue del sistema operativo en los nodos de almacenamiento

Con el servidor PXE descrito ya desplegado en el clúster, podemos comenzar a realizar la instalación del sistema operativo en los nodos que formarán la nueva infraestructura de almacenamiento.

Únicamente es necesario arrancar el nodo y seleccionar como dispositivo de arranque una tarjeta de red compatible con PXE y conectada a la red de comunicación del clúster, donde tenga acceso al servicio DHCP y al servidor PXE.

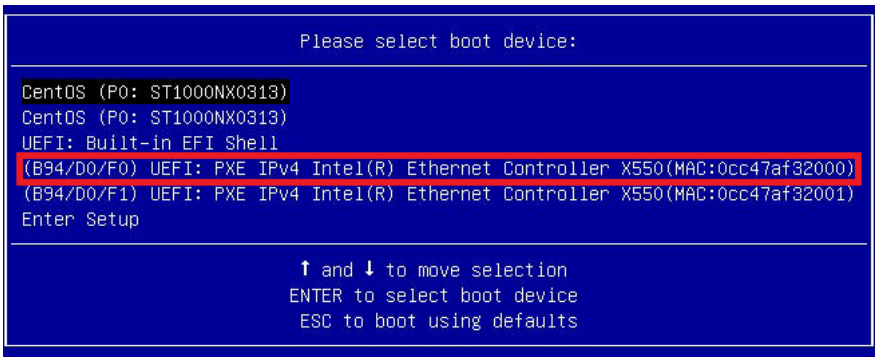


Ilustración 9: Menú de arranque de un nodo del clúster.

En la ilustración 9 se observa un ejemplo de un menú de arranque de uno de los nodos del clúster, donde se destaca la opción que corresponde con una tarjeta de red compatible con el protocolo PXE.

Tras seleccionar el medio de arranque correcto, el nodo comienza la comunicación con el servidor PXE y realiza todos los pasos hasta alcanzar por último el servidor FTP, donde se encuentra la imagen de instalación del sistema operativo para el nodo. Una vez llegado a este punto, comienza la ejecución del programa de instalación (Anaconda[22] en el caso de RedHat) que completará el proceso de instalación de forma completamente desatendida haciendo uso del fichero kickstart proporcionado junto con la fuente de instalación.

[22] https://docs.centos.org/en-US/centos/install-guide/Graphical_Installation-ppc/

## 4.3 Despliegue de la configuración y servicios de los nodos de almacenamiento

Una vez instalado el sistema operativo inicial, comienza el proceso de instalación y/o configuración del software y servicios necesarios para que el nodo desempeñe su papel en la nueva infraestructura de almacenamiento de altas prestaciones. En este apartado se describen los principales componentes del despliegue, como son el sistema de ficheros raíz empleado para el sistema operativo, el software necesario para la integración del nodo en la infraestructura de inicio único de sesión (SSO) y directorios de trabajo remotos de los usuarios presentes en el clúster. Finalmente, se completa la configuración del nodo con el despliegue del sistema de ficheros paralelo y distribuido Lustre.

### 4.3.1 Sistema de ficheros ZFS

Creado por Sun Microsystems en 2001, para el sistema operativo Solaris, y más tarde con el lanzamiento de OpenSolaris en noviembre de 2005, bajo licencia de código abierto CDDL[23]. Durante los siguientes años esta versión de código abierto fue migrada a diversos sistemas operativos, incluidas diversas distribuciones Linux. En 2013 se creó el proyecto OpenZFS[24] con el fin de coordinar todo este desarrollo de ZFS como código abierto.

Vamos a comentar, sin entrar en detalle sobre el funcionamiento interno e implementación debido a su complejidad, algunas de las principales características que hacen de ZFS un sistema de ficheros seguro, altamente escalable, fácil de administrar, que además ofrece un alto rendimiento y con capacidad para organizar y gestionar diferentes unidades bajo un mismo espacio de almacenamiento [12][13].

#### Sistema de ficheros basado en pool de discos

Históricamente, los sistemas de ficheros se han construido sobre un único dispositivo físico. Posteriormente, para agregar múltiples dispositivos de almacenamiento, se introdujo el concepto de gestor de volúmenes lógicos (LVM). Este diseño agregó una nueva capa de complejidad entre el sistema de ficheros y los dispositivos físicos de almacenamiento, limitando al sistema de ficheros de la capacidad de controlar el almacenamiento físico de los datos [7].

Sin embargo, ZFS es fundamentalmente diferente en este aspecto, porque es más que un sistema de ficheros. ZFS combina las funciones de un sistema de ficheros y de un gestor de volúmenes lógicos. De esta forma, ZFS puede controlar de manera explícita y más eficiente los aspectos relativos a la gestión y organización de los dispositivos físicos de almacenamiento (discos) , introduciendo adicionalmente a LVM, redundancia y tolerancia a fallos hardware.

[23] https://opensource.org/licenses/CDDL-1.0
[24] https://openzfs.org

ZFS agrega dispositivos (i.e. discos) en un grupo de almacenamiento (pool). Este pool describe las características físicas del almacenamiento (parámetros de rendimiento, redundancia de datos, etc.) y actúa como un almacén de datos arbitrario a partir del cual se pueden crear varios sistemas de ficheros (dataset). Esto permite agregar dispositivos de almacenamiento adicionales a un sistema activo y tener el nuevo espacio disponible en todos los sistemas de archivos existentes en dicho pool, de forma inmediata y sin interrumpir la actividad del sistema (en caliente).

### Snapshots

ZFS implementa de forma nativa un sistema de instantáneas (snapshots). Cuando se crea un snapshot de un dataset, este contiene el estado del dataset original en el momento de su creación, y va almacenando solamente los cambios realizados desde que su creación. Sólo se utiliza espacio a medida que se escriben nuevos datos en el dataset, asignándose nuevos bloques para almacenar estos datos.

Los snapshots se pueden usar para recuperar una versión anterior de un archivo, aunque también es posible revertir el sistema de ficheros completo a una instantánea anterior, o incluso crear un dataset nuevo (clone), de forma que sólo se almacenan las diferencias entre el dataset original y el clone. Para más detalles sobre la administración de un sistema ZFS ver el Capítulo 19 del Manual de FreeBSD ([13]).

### RAID-Z

ZFS implementa su propia funcionalidad RAID. Denominada RAID-Z ,además de manejar y soportar el fallo hardware de un disco completo, también puede detectar y corregir los fallos que se introducen cuando se realiza una operación de lectura o escritura sobre el sistema de ficheros, ofreciendo una corrección automática. Cuando se lee un bloque RAID-Z, ZFS comprueba su suma de comprobación, y si no devuelve la respuesta correcta, determina el disco donde está localizado el error y repara los datos dañados.

Hay varios niveles RAID-Z diferentes: Striped Vdev's (similar a RAID 0, no ofrece redundancia), Mirrored Vdev's (similar a RAID 1, permite el fallo de todos los discos menos 1), RAID-Z1 (similar a RAID 5, permite el fallo de un disco), RAID-Z2 (similar a RAID 6, permite el fallo de 2 discos), RAID-Z3 (una configuración RAID 7, permite el fallo de tres discos).

### Integridad de datos y reparación automática

ZFS implementa un sistema de detección y recuperación automática de inconsistencias en los datos almacenados. La inconsistencia se puede descubrir usando las sumas de verificación (checksums) almacenadas para cada bloque del dataset, y su recuperación se basa en la redundancia de datos del RAIDZ o su duplicidad.

En conclusión, ZFS cuenta con un gran número de características que facilitan la creación, modificación y mantenimiento del sistema de ficheros, además de proporcionar herramientas para garantizar la integridad y recuperación de los datos almacenados. Es por todo ello por lo que ZFS es considerado actualmente como una de las opciones principales para la gestión de los sistemas de almacenamiento, en entornos de la computación y almacenamiento de altas prestaciones y alta disponibilidad.

Sin embargo, ZFS presenta un inconveniente a la hora de instalar el sistema de ficheros en los nodos de un clúster. ZFS está distribuido bajo una licencia de tipo CDDL y por tanto incompatible con la licencia GPL[25] del Kernel de Linux, esto prohíbe la distribución del software ZFS junto con el Kernel de linux, lo que significa que no es posible incluir el sistema de ficheros ZFS en una imagen de instalación de un sistema operativo Linux, por lo que no está disponible durante el proceso de instalación del sistema operativo en los nodos. Esta limitación no impide que un tercero desarrolle y distribuya un módulo de kernel de Linux nativo, como es el caso de ZFSonLinux.

Por este motivo, ZFS no puede ser configurado por defecto, como sistema de ficheros de arranque (boot) o raíz del sistema operativo Linux, durante el proceso de instalación del propio sistema operativo. En el apartado 4.2.2 explicaremos el proceso necesario para completar el despliegue de ZFS en los nodos del clúster como sistema de ficheros raíz.

#### 4.3.1.1 Diseño de la configuración

Hemos comenzado exponiendo los objetivos y características de los sistemas de ficheros ZFS que queremos implementar en la infraestructura de almacenamiento del clúster. En concreto, se pretenden desplegar dos sistemas de ficheros diferentes en cada uno de los nodos de almacenamiento; un sistema de ficheros para el sistema operativo y otro para el backend del sistema de almacenamiento de altas prestaciones.

- **Sistema de ficheros raíz:** Este sistema de ficheros contiene el sistema operativo instalado en los nodos. El requisito principal que planteamos para este sistema de ficheros del sistema es la tolerancia al fallo de un dispositivo de almacenamiento. Para esto usaremos el mínimo número de discos necesarios (dos) formando un pool de almacenamiento de tipo mirror (equivalente a RAID 1). De esta forma el fallo de uno de los dos discos no supone una interrupción del servicio de los nodos. Con esta configuración, en el caso de que falle uno de los discos del sistema, el dispositivo defectuoso podrá ser reemplazado por uno nuevo y el software de ZFS volverá a copiar, de forma automática, todos los datos en el nuevo disco para recuperar la configuración mirror, todo esto sin necesidad de apagar el sistema o interrumpir el servicio.

- **Sistema de ficheros de altas prestaciones Lustre:** Este sistema de ficheros, como explicaremos en el apartado 4.4, proporcionará el backend de almacenamiento utilizado para el despliegue del sistema de ficheros Lustre. Por el momento basta con entender los requisitos que deseamos tenga este sistema de ficheros. En este caso,

[25]https://en.wikipedia.org/wiki/GNU_General_Public_License

buscamos el mejor balance posible entre rendimiento, eficiencia espacial y seguridad e integridad de los datos.

#### 4.3.1.2 Despliegue

Como ya mencionamos en la introducción de este capítulo, el software de ZFS no puede ser distribuido junto con el kernel de Linux, por lo que para realizar el despliegue completo de los sistemas de ficheros raíz de los nodos, será necesario seguir el proceso descrito a continuación [7][14]:

1. Para realizar el despliegue de ZFS, como sistema de ficheros raíz en un nuevo nodo, debemos realizar primero una instalación del sistema operativo sobre una unidad de almacenamiento formateada con un sistema de ficheros raíz compatible disponible desde el programa de instalación (EXT3, EXT4, XFS, ...), a través de nuestro servidor PXE. Es importante a la hora de instalar el sistema operativo, no utilizar (dejar libre) al menos uno de los dos dispositivos de almacenamiento reservados para la creación del sistema de ficheros raíz ZFS del nodo.

2. Una vez instalado el sistema operativo inicial sobre un sistema de ficheros raíz compatible disponible (EXT3, EXT4, XFS, ...), se procederá a la creación del sistema ZFS y a la migración del sistema operativo al nuevo sistema. Para ello se han creado dos scripts que llevan a cabo el resto de las tareas de forma automática. A continuación, se exponen los pasos de forma simplificada. Si se desea un mayor grado de detalle, en los anexos de este documento se muestran los scripts en su totalidad.

   El motivo por el que se desarrollan dos scripts es que, en cierto punto, es necesario reiniciar el sistema para poder continuar con el despliegue del sistema de ficheros ZFS raíz. El primer script realiza las siguientes tareas:

   1. Se descarga todo el software de ZFS necesario[26].

   2. Crear el pool para el nuevo sistema de ficheros raíz. En nuestro caso es un mirror pool, sin embargo, por el momento se crea un pool con un solo dispositivo de almacenamiento, con uno de los discos reservados para el sistema de ficheros raíz y que no está siendo usado por el sistema de ficheros del sistema operativo inicial.

   3. Migrar (copiar) el sistema de ficheros original al nuevo sistema de ficheros raíz ZFS.

   4. Crear las particiones de arranque e instalar y configurar el bootloader (GRUB2) necesario para el arranque.

   Tras este paso es necesario reiniciar el sistema y arrancar el nuevo sistema ZFS raíz,

[26] http://download.zfsonlinux.org/epel/zfs-release.el7_7.noarch.rpm

de esta forma se puede utilizar el disco de la instalación original para formatearlo y añadirlo al pool raíz. A continuación, se ejecuta el segundo script, que finaliza el despliegue del sistema de ficheros raíz ZFS:

1. Añadir el disco del sistema de ficheros original al pool raíz en configuración mirror (RAID 1).

2. Crear las particiones de arranque y configura el bootloader (GRUB2) necesario para el arranque desde el segundo disco.

Tras completar el despliegue del nuevo sistema de ficheros ZFS raíz en los nodos, el siguiente paso es crear el pool de almacenamiento para el sistema de ficheros Lustre. Sin embargo, este pool no es idéntico en todos los nodos ya que su estructura de almacenamiento será diferente en función del propósito que para cada uno de los nodos establezca la arquitectura Lustre, como se expondrá en el apartado 4.3.4. Por este motivo, el despliegue del sistema de ficheros Lustre en los nodos se recoge en el apartado 4.3.4.4, dedicado íntegramente al despliegue del sistema de ficheros Lustre en el clúster.

#### 4.3.1.3 Integración en el aprovisionamiento automático

Los scripts desarrollados, descritos en el apartado anterior, realizan el proceso completo de instalación y configuración del software ZFS de forma automática. Sin embargo, queremos integrar este proceso en la infraestructura de aprovisionamiento automático, de manera que el despliegue de este componente se realice a través de la herramienta Ansible, al igual que el resto de los componentes del despliegue descritos con anterioridad.

Como es habitual, se ha creado un nuevo y sencillo *playbook* que automatiza todos los pasos del despliegue del sistema de ficheros raíz ZFS:

1. Copia los dos scripts y archivos de configuración en el host remoto.
2. Ejecuta el primer script.
3. Reinicia el host remoto.
4. Ejecuta el segundo script.

De esta forma el proceso de despliegue se realiza con la ejecución de un solo *playbook* desde el nodo de control.

### 4.3.2 Identificación y autentificación de usuarios

Este apartado describe el proceso necesario para integrar los nuevos nodos de almacenamiento en la infraestructura de Inicio Único de Sesión (SSO) presente en el clúster del servicio de Datos Climáticos, descrita en el apartado 3.3.1.

Si recordamos, la infraestructura SSO presente en el clúster actual, consiste en un servidor de Directorio Activo, que utilizar el protocolo de autenticación Kerberos.

#### 4.3.2.1 Despliegue e Integración en el aprovisionamiento automático

Para garantizar el correcto despliegue e integración en la infraestructura de aprovisionamiento automático, seguimos los mismos pasos que con el resto de los componentes [7]:

- **Definición del objetivo**: integrar el nodo en el dominio de inicio único de sesión presente en el clúster.

- **Métodos o herramientas a utilizar**: para determinar el software a emplear y su configuración, nos apoyamos en la documentación oficial que proporciona RedHat para este caso de uso [15][16]. Concretamente, se utilizarán los siguientes componentes:

  System Security Services Daemon (SSSD[27]); Se trata de es un servicio del sistema cuya principal función es garantizar acceso a directorios remotos y mecanismos de autenticación. Conecta un sistema local (un cliente SSSD) con un sistema back-end externo (un dominio). Esto proporciona al cliente SSSD acceso a servicios remotos de identidad y autenticación.

  realmd[28]: Herramienta software que proporciona una forma clara y sencilla de descubrir y unirse a dominios SSO. Configura los servicios del sistema Linux subyacentes, como SSSD, de forma automática.

- **Despliegue y pruebas**: una vez instalado el software, el proceso de configurar un nodo mediante la herramienta realmd puede ser tan sencillo como ejecutar el siguiente comando: `realm join -U admin MACC.UNICAN.ES.`

  Este comando añade el nodo al dominio SSO (MACC.UNICAN.ES) tras introducir las credenciales de un usuario (admin) autorizado en el servidor AD.

- **Integración en el aprovisionamiento automático:** para automatizar el despliegue, se ha creado un *playbook* cuyas principales funciones son instalar el software necesario, ejecutar el comando “realm join” y reiniciar el servicio sssd en el nodo.

[27] https://sssd.io/
[28] https://www.freedesktop.org/software/realmd/

### 4.3.3 Directorio de trabajo remoto de los usuarios

Este apartado recoge el proceso de despliegue seguido para integrar el servicio de directorios de trabajo remotos de los usuarios, descrito en el apartado 3.3.2, en los nuevos nodos de almacenamiento.

Si recordamos, este servicio está implementado mediante un sistema de ficheros, accesible desde cualquier nodo a través de la red interna del clúster, mediante el protocolo NFS.

Para garantizar el correcto despliegue e integración en la infraestructura de aprovisionamiento automático, seguimos los mismos pasos que con el resto de los componentes [7]:

- **Definición del objetivo**: proporcionar al nuevo nodo acceso a la infraestructura de directorios de trabajo remotos. Adicionalmente, se requiere que el nodo disponga de acceso a esta infraestructura de forma dinámica, es decir, únicamente hará uso de este sistema de ficheros remoto cuando sea necesario (un usuario acceda a su directorio de trabajo).

  Un inconveniente de montar el sistema de ficheros de forma estática (/etc/fstab) es que, independientemente de la frecuencia con la que un usuario acceda a su directorio de trabajo remoto, montado mediante NFS, el servidor debe dedicar recursos para mantener el sistema de ficheros montado permanentemente. Cuando el servicio mantiene muchos sistemas de ficheros montados al mismo tiempo, el rendimiento general del servicio puede verse afectado. Una alternativa a fstab es usar la utilidad automount, que permite montar y desmontar sistemas de ficheros de forma dinámica bajo demanda [17].

- **Métodos o herramientas a utilizar**: para cumplir estos requisitos, el nodo debe contar con el software de la parte del cliente del protocolo NFS (nfs-utils) y una herramienta para gestionar la utilidad automount (autofs[29]).

- **Despliegue y pruebas**: una vez instalado el software, es necesario editar los ficheros de configuración de la herramienta autofs para indicar la localización del sistema de ficheros NFS que contiene los directorios de trabajo de los usuarios.

- **Integración en el aprovisionamiento automático**: para automatizar el despliegue, se ha creado un *playbook* cuyas principales funciones son instalar el software necesario, copiar los ficheros de configuración (almacenados en el nodo maestro) en el nodo de destino y habilitar y reiniciar el servicio autofs.

[29] https://wiki.archlinux.org/index.php/Autofs

### 4.3.4 Sistema de ficheros Lustre

Por último, tras haber completado el despliegue de todos los anteriores componentes, llegamos a la que será la última pieza necesaria para completar el aprovisionamiento de los servidores que componen la nueva infraestructura de almacenamiento del clúster de Servicios de Datos Climáticos de la Universidad de Cantabria.

#### 4.3.4.1 ¿Qué es?

Lustre es una plataforma software de código abierto que permite el despliegue de un sistema de ficheros paralelo y distribuido diseñado específicamente para ofrecer una alta escalabilidad, rendimiento y disponibilidad.

Lustre se ejecuta en sistemas operativos basados en Linux y emplea una arquitectura cliente-servidor. El almacenamiento es proporcionado por un conjunto de servidores escalables hasta varios cientos de hosts. Los servidores Lustre, para una sola instancia del sistema de ficheros pueden presentar, en agregado, hasta decenas de petabytes de almacenamiento a miles de clientes simultáneamente, y proporcionar más de un terabyte por segundo de rendimiento combinado [18].

#### 4.3.4.2 Arquitectura Lustre

Lustre presenta un almacenamiento distribuido basado en dos tipos de objetos, que se almacenan por separado en servidores diferentes. Por un lado, se almacena la jerarquía del espacio de nombres o inodos (metadatos), y por otro el contenido de los archivos (datos) [18][19].

Los servidores en Lustre están separados en aquellos que admiten operaciones sobre los metadatos o inodos y aquellos que admiten operaciones sobre el contenido de los archivos o datos. También existe un servidor de administración, encargado específicamente de la gestión de un registro global de información sobre la configuración de todo el espacio Lustre, el cual es funcionalmente independiente de cualquiera de las otras instancias del sistema de ficheros.

Cada servidor tiene al menos un dispositivo de almacenamiento denominado *target,* donde se almacena la información necesaria para desempeñar su papel dentro del sistema de ficheros Lustre.

- **Servidores de Metadatos (MDS):** almacena metadatos del espacio de nombres, como nombres de archivos, directorios y permisos de acceso, proporcionando de manera efectiva el índice de los datos contenidos en el sistema de archivos. Además controla la asignación de nuevos objetos de almacenamiento en los servidores de almacenamiento para los datos cuando se crea un archivo nuevo y gestiona las operaciones sobre los metadatos de los archivos como la apertura y cierre de archivos, eliminaciones y cambios de nombre y otras operaciones sobre los inodos.

La información de metadatos del sistema Lustre está contenida en los dispositivos de almacenamiento denominados Metadata Targets (MDT), los cuales proporcionan la interfaz lógica necesaria para la gestión para esta información. Un sistema de archivos Lustre siempre tendrá al menos un MDS y el MDT correspondiente, y se pueden agregar más para cumplir con los requisitos de escala de un entorno particular.

- **Servidores de Almacenamiento de Objetos (OSS):** proporciona almacenamiento masivo para el contenido de los archivos. Uno o más servidores de almacenamiento de objetos (OSS) almacenan los datos de los archivos en uno o más Object Storage Target (OST). Un solo sistema de ficheros Lustre puede escalar a cientos de OSS.

  La capacidad de un sistema de ficheros Lustre es la suma de las capacidades proporcionadas por los OST en todos los hosts OSS.

- **Servidor de administración (MGS):** almacena la configuración para todos los sistemas de ficheros de Lustre en un clúster y proporciona esta información a otros hosts. Los servidores y los clientes se conectan al MGS en el inicio para recuperar el registro de configuración del sistema de ficheros. El MGS registra la información de configuración en un dispositivo de almacenamiento llamado Management Target (MGT).

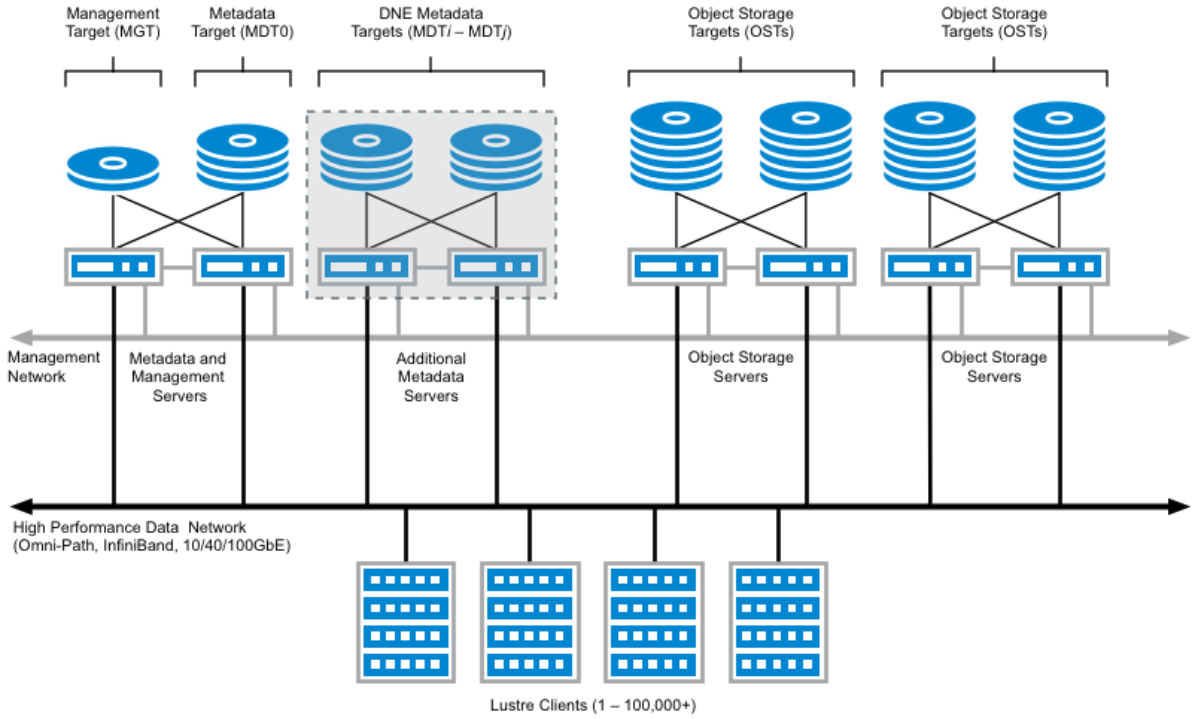


Ilustración 10: Arquitectura del sistema de ficheros Lustre. Fuente: wiki.lustre.org

En la ilustración 10 se puede observar un ejemplo de la arquitectura estándar de un sistema de ficheros Lustre, donde se muestran todos los componentes de la arquitectura descritos anteriormente. De izquierda a derecha, comenzando por la parte superior, se observa el servidor de administración (MGS) y los servidores de metadatos (MDS) con sus respectivos

MGT y MDTs. En la parte derecha de la ilustración, se observan los servidores OSS con sus respectivos OSTs.

En la ilustración también están representadas la red de configuración y la red de altas prestaciones para la comunicación entre los servidores y los clientes.

#### 4.3.4.3 Diseño de la arquitectura Lustre en el clúster

Previo al comienzo del desarrollo de este trabajo de fin de grado, el Grupo de Meteorología y Computación del Departamento de Matemática Aplicada y Ciencias de la Computación de la Universidad de Cantabria, ya había realizado la compra e instalación en el clúster de las máquinas destinadas a la nueva infraestructura de almacenamiento Lustre.

En total, esta ampliación proporciona 14 nuevos servidores, 12 con características pensadas para ejercer la función de servidores de almacenamiento de objetos (OSS) y 2 para servidores de metadatos y de configuración (MDS y MGS).

La principal diferencia entre ambos tipos de servidores está en las características de almacenamiento. Cada nodo OSS dispone de una bahía con capacidad para 60 discos SATA3[30]. En cambio, los dos nodos MDS comparten una bahía con capacidad para 12 discos SAS3[31], accesibles por doble canal.

**Servidores de almacenamiento de objetos (OSS01 - OSS12)**

Según la arquitectura de Lustre por definición, estos servidores proporcionan el almacenamiento masivo para el contenido de los archivos. Esto hace que la configuración de estos equipos sea determinante en el rendimiento, integridad y capacidad del sistema de ficheros.

Por estos motivos, como se hizo referencia en el capítulo 4.2, se emplea el sistema de ficheros ZFS como *backend* para el pool de almacenamiento en cada servidor OSS.

**Recomendaciones para el almacenamiento OST**

Esta sección describe pautas y recomendaciones con respecto al *backend* de almacenamiento, extraídas del manual de operaciones de Lustre 2.x [20].

*"Se necesita RAID 6 u otro algoritmo de doble paridad para proporcionar suficiente redundancia para el almacenamiento OST. Para un mejor rendimiento, le recomendamos que cree conjuntos RAID con 4 u 8 discos de datos más uno o dos discos de paridad. Usar conjuntos RAID más grandes afectará negativamente el rendimiento en comparación con tener múltiples conjuntos RAID independientes.*

[30] https://en.wikipedia.org/wiki/Serial_ATA
[31] https://en.wikipedia.org/wiki/Serial_Attached_SCSI

*Para maximizar el rendimiento de los pequeños tamaños de solicitud de E/S, el almacenamiento configurado como RAID 1+0 puede producir mucho mejores resultados, pero aumentará el coste o reducirá la capacidad."* [20]

En cuanto a la capacidad de cada OST, la decisión es tomada en función de las necesidades de almacenamiento del sistema de ficheros. En este caso, se ha optado por asignar 30 discos de 5TB de capacidad a cada OST, dejando libres los otros 30 espacios para futuras expansiones.

En la siguiente ilustración se muestra el esquema de la configuración de los servidores OSS.

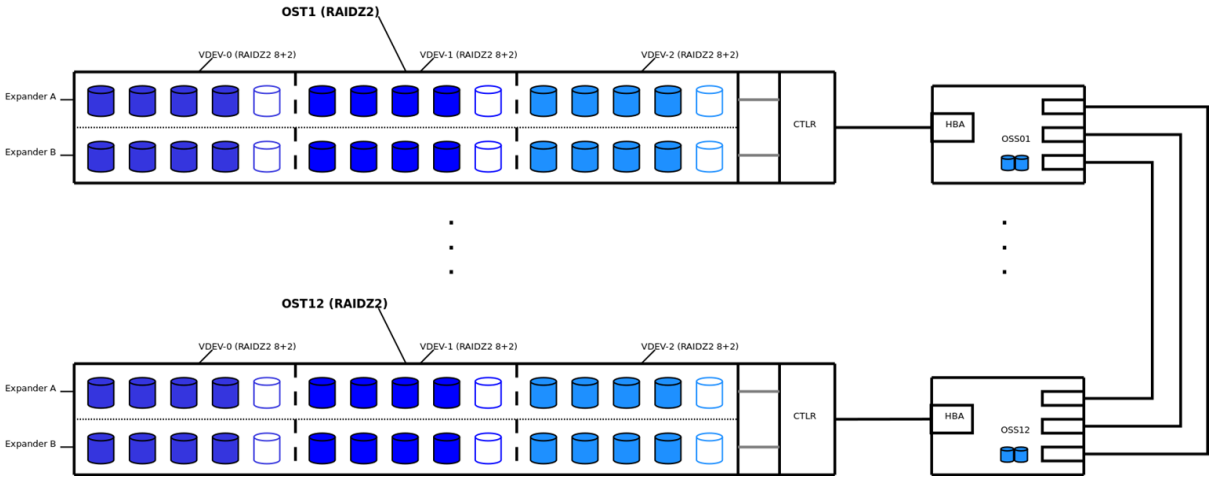


Ilustración 11: Configuración de los servidores OSS del sistema de ficheros Lustre desplegado en el CPD.

Se observa que cada OST, siguiendo las pautas del manual, está compuesto por 3 conjuntos RAID 6 independientes (VDEVS), cada uno formado por 8 discos más 2 de paridad.

Con esta configuración logramos un almacenamiento disponible de aproximadamente 120 TB por OST (3 VDEVS RAID 6 (8 + 2) con discos duros de 5 TB), el cual agrega hasta 1.4 PB de almacenamiento total entre los 12 OSS.

**Servidor de metadatos y Servidor de administración (MDS y MGS)**

Según la arquitectura de Lustre por definición, los servidores de metadatos almacenan el conjunto de *inodos* del sistema de ficheros y juegan un papel fundamental en el acceso a los éstos. Por lo tanto, la configuración de estos equipos es también determinante en el rendimiento e integridad del sistema de ficheros.

Por estos motivos, al igual que para los OSTs, se emplea el sistema de ficheros ZFS como *backend* para el pool de los servidores de metadatos, considerando para su implementación las pautas y recomendaciones expuestas en el manual de operaciones de Lustre 2.x [20].

Por otra parte, el servicio MGS almacena la configuración del sistema de ficheros Lustre y proporciona esta información a los hosts. Los servidores y los clientes solo se conectan al MGS en el inicio para recuperar el registro de configuración del sistema de ficheros, por lo

que este servicio no influye en el rendimiento del sistema de ficheros.

**Recomendaciones para el almacenamiento MDT y MGT**

*"La E/S en el MDT suele ser principalmente lectura y escritura de pequeñas cantidades de datos (inodos). Por esta razón, recomendamos que use RAID 1 para el almacenamiento MDT. Si necesita más capacidad para un MDT, recomendamos RAID 1 + 0 o RAID 10."* [20]

La capacidad de almacenamiento del servidor MDS no es una decisión trivial. El almacenamiento MDT debe ser lo suficientemente grande como para almacenar los *inodos* de todos los archivos del sistema de ficheros, y por lo tanto depende de la capacidad total de éste.

En el almacenamiento MGT, tanto la capacidad como el rendimiento son aspectos triviales. La única recomendación a tener en cuenta es elegir una configuración que garantice la integridad y recuperación de los datos ante el fallo en un dispositivo de almacenamiento. En nuestro caso utilizamos una configuración RAID 1 con dos discos.

Teniendo en cuenta estas recomendaciones, se ha determinado el diseño para nuestro MDT de la siguiente manera:

Para estimar la capacidad necesaria para el almacenamiento MDT, calculamos el número total de *inodos* mínimo. En nuestro caso, la capacidad total de almacenamiento de objetos es de aproximadamente de 1.4 PB y se estima un tamaño promedio del archivo de 5 MB. Así, el número mínimo de *inodos* se puede calcular dividiendo el almacenamiento OST total por el tamaño promedio del archivo:

1.4 PB / 5 MB por *inodo* = 2.8x10^8 *inodos*

Para permitir una expansión futura o un permitir el ajuste del tamaño de archivo promedio a un tamaño más pequeño de lo esperado, el espacio debe reservarse para el doble del número mínimo de *inodos*. Dado que cada *inodo* puede agregar hasta 2 KB de espacio, el almacenamiento de metadatos debe ser al menos el siguiente:

2 KB por inodo x 2.8x10^8 inodos x 2 ≈ **1.14 TB** de almacenamiento MDT

En la siguiente ilustración se muestra el esquema de la configuración de los servidores MDS y MGS.

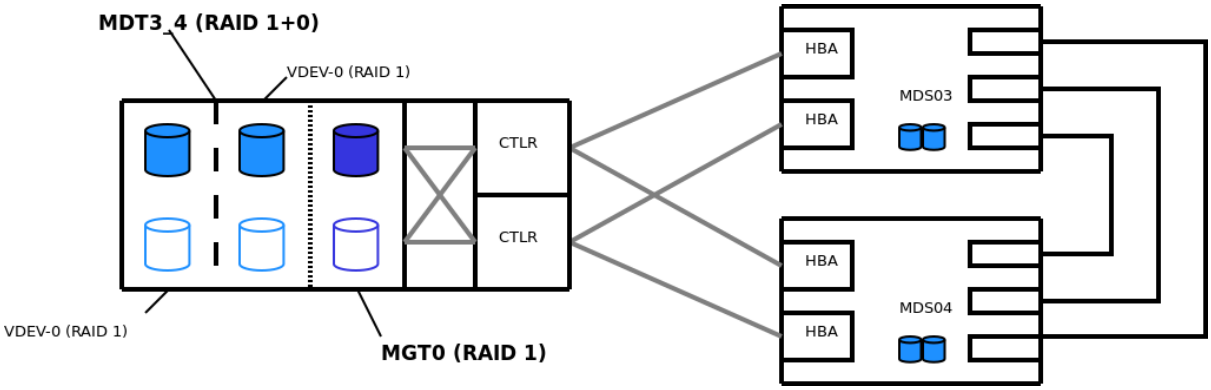


Ilustración 12: Configuración de los servidores MDS y MGS del sistema de ficheros Lustre desplegados en el CPD.

Como se menciona en el apartado 4.5.3 (Diseño de la arquitectura Lustre en el clúster), los dos servidores de metadatos tienen acceso de forma simultánea al almacenamiento compartido MGT y MDT. Esto permite que uno de los equipos ejecute el servicio MGS y el otro el servicio MDS en configuración de *conmutación por error*, permitiendo a cualquiera de los servidores asumir ambos servicios en el caso de que se produzca un fallo en el otro servidor.

Siguiendo las pautas del manual y teniendo en cuenta el requisito de capacidad mínima calculado, el MDT estará implementado por un zpool RAID 1+0 compuesto por dos VDEV RAID 1, con una capacidad resultante de cerca de 3 TB de almacenamiento, bastante superior a los 1.14 TB estimados.

El MGT está implementado por un VDEV RAID 1 de dos discos, garantizando la integridad y recuperación de los datos en caso de fallo en uno de los discos.

### 4.3.4.4 Despliegue

El proceso de instalación y configuración del software Lustre en los servidores del clúster es relativamente sencillo. Primero, se procede a la instalación del software Lustre necesario, procedente de un repositorio online[32] o local. A continuación, usando las herramientas software que se han instalado, se crea el target o targets de almacenamiento y se inicia el servicio Lustre correspondiente (MGS, MDS o OSS) en cada servidor.

A continuación, se expone cada uno de los pasos necesarios para completar el despliegue del sistema de ficheros Lustre en el clúster [19][20].

#### Instalación del software Lustre

El software Lustre puede ser instalado desde paquetes (RPM) o directamente desde el código fuente. En nuestro caso, la instalación se realiza mediante paquetes RPM, procedentes de un repositorio local, sincronizado con uno de los repositorios oficiales de Lustre. De esta

[32] https://downloads.whamcloud.com/public/lustre/

forma, se puede controlar más eficazmente la auditoría y actualización del software Lustre instalado en cualquiera de los servidores del clúster.

Siguiendo los pasos expuestos en el manual de operaciones de Lustre, se instalan los paquetes necesarios (los mismos para todos los servidores) en todos los nodos que forman la nueva infraestructura de almacenamiento Lustre.

Una vez instalado el software, se realiza el proceso de creación y configuración de los servicios MGS, MDS y OSS.

#### Creación del target de almacenamiento (MGT, MDT o OST)

Siguiendo el diseño realizado para cada uno de los targets, se utiliza la herramienta zpool[33] para, previamente, crear un pool de almacenamiento ZFS que sirva como *backend* para el sistema de ficheros Lustre que se creará a continuación .

```
[padillad@mds01 ~]$ zpool create pool_mgt mirror /dev/mapper/mpath-bay01-358ce38
ee2095abed /dev/mapper/mpath-bay13-358ce38ee2095ab85
```

Ilustración 13: comando usado para la creación del target MGT.

#### Creación del sistema de ficheros Lustre

Se emplea la herramienta mkfs.lustre[34] para crear el sistema de ficheros Lustre dentro del pool de almacenamiento ZFS creado en el paso anterior.

```
[padillad@mds01 ~]$ mkfs.lustre --mgs --backfstype=zfs --fsname=lustre01 pool_mg
t/mgt
```

Ilustración 14: comando usado para la creación del sistema de ficheros Lustre del servicio MGS.

#### Inicio del servicio (MGS, MDS o OSS)

Por último, se inicia el servicio Lustre con el montaje del sistema de ficheros creado en el nodo.

```
[padillad@mds01 ~]$ mount -t lustre pool_mgt/mgt /mnt/mgt
```

Ilustración 15: comando usado para iniciar el servicio MGS en el nodo.

### 4.3.4.5 Integración en el aprovisionamiento automático

En el momento de la escritura de esta memoria, aún no se ha realizado la integración del sistema de ficheros Lustre en la infraestructura de aprovisionamiento automático desarrollada a lo largo del TFG. Por motivos de tiempo, no ha sido posible completarlo para

[33] https://linux.die.net/man/8/zpool
[34] http://manpages.ubuntu.com/manpages/precise/man8/mkfs.lustre.8.html

ser incluido en esta memoria. Sin embargo, de forma similar al resto de los componentes del despliegue, se realizará próximamente el desarrollo de los *playbooks* necesarios para integrar el despliegue del sistema de ficheros Lustre en la infraestructura Ansible.

# 5. Conclusiones y líneas futuras

El objetivo de este proyecto ha sido el despliegue, integración y aprovisionamiento automático de una nueva infraestructura de almacenamiento de altas prestaciones en el clúster del Servicio de Datos Climáticos del Grupo de Meteorología y Computación, perteneciente al Departamento de Matemática Aplicada y Ciencias de la Computación de la Universidad de Cantabria.

Una gran parte del trabajo realizado ha consistido en diseñar y desplegar los componentes necesarios para la creación de una infraestructura de almacenamiento de estas características y su integración en un entorno ya definido. Aunque es cierto que esta nueva infraestructura aún se encuentra en estado de desarrollo, se espera que su uso en producción esté disponible para los usuarios del clúster en los próximos meses.

Al mismo tiempo, haciendo uso de la herramienta Ansible, se ha desarrollado una nueva plataforma software para el aprovisionamiento automático, compuesta por decenas de *playbooks,* muchos de ellos desarrollados en este proyecto, capaces de reconstruir de forma desatendida el estado de los diferentes componentes desplegados en los diferentes nodos del clúster. Esta nueva infraestructura software supone una herramienta muy útil para el mantenimiento y administración de los nodos y servicios del clúster, así como para la incorporación de nuevos servidores en el futuro.

En conclusión, el trabajo realizado satisface en gran medida los objetivos planteados al comienzo de este TFG. Aunque quizás, lo más destacable sea el impacto que las nuevas infraestructuras de almacenamiento implementadas y el aprovisionamiento automático desarrollado, van han suponer para los usuarios y administradores del clúster respectivamente.

En cuanto a las competencias técnicas empleadas durante la realización de este TFG, ha sido imprescindible poner en práctica los conocimientos adquiridos durante el grado, en materias relacionadas con la arquitectura y redes de computadores y por supuesto, con la administración de sistemas. Además, ha sido necesario adquirir nuevos conocimientos, destrezas y competencias en estas áreas. Pero, uno de los mayores desafíos y experiencias adquiridas ha sido sin duda, incorporarse al entorno de trabajo de un clúster de computadores de altas prestaciones y tener un contacto directo sobre todos sus componentes (hardware y software) y sobre la administración de toda una infraestructura de estas características.

Por último, desde el punto de vista personal, el trabajo realizado ha sido sin duda una experiencia muy positiva, que me ha permitido adquirir experiencia trabajando en un entorno profesional, donde ha sido necesaria pero también gratificante la interacción con otros miembros del grupo para la realización de este trabajo.

En lo que respecta al futuro de este proyecto, el despliegue de una infraestructura de estas características en un entorno de trabajo como es el clúster del Servicio de Datos Climáticos requiere el mantenimiento y optimización (hardware y software) de todos los componentes de cara a garantizar el rendimiento y la disponibilidad a lo largo de la vida útil de la nueva infra-estructura.

# 6. Referencias


[1] «The National Institute for Computational Sciences | What is HPC». [En línea]. Disponible en: https://www.nics.tennessee.edu/computing-resources/what-is-hpc.

[2] «Iowa State University | High Performance Computing». [En línea]. Disponible en: https://www.hpc.iastate.edu/guides/introduction-to-hpc-clusters/what-is-an-hpc-cluster#

[3] «Digital Ocean | High Availavitlity». [En línea]. Disponible en: https://www.digitalocean.com/community/tutorials/what-is-high-availability.

[4] «Wikipedia | High Availability». [En línea]. Disponible en: https://en.wikipedia.org/wiki/High_availability.

[5] «RedHat | Provisioning». [En línea]. Disponible en: https://www.redhat.com/en/topics/automation/what-is-provisioning#:~:text=Server%20provisioning%20is%20the%20process,desired%20state%20of%20the%20system.

[6] L. Hochstein, *Ansible: Up and Running*, 1 edition. Beijing: O'Reilly Media, 2015.

[7] «UNIX and Linux System Administration Handbook (5th Edition): Nemeth, Evi, Snyder, Garth, Hein, Trent R., Whaley, Ben, Mackin, Dan: 9780134277554: Amazon.com: Books». https://www.amazon.com/UNIX-Linux-System-Administration-Handbook/dp/0134277554.

[8] J. De Clercq, «Single Sign-On Architectures», en *Infrastructure Security*, vol. 2437, G. Davida, Y. Frankel, y O. Rees, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 40-58.

[9] «OpenStack Docs: Bare Metal Service User Guide». https://docs.openstack.org/ironic/latest/user/index.html.

[10] «pxespec.pdf». [En línea]. Disponible en: http://www.pix.net/software/pxeboot/archive/pxespec.pdf.

[11] «Preparing for a Network Installation». [En línea]. Disponible en: https://docs.centos.org/en-US/centos/install-guide/pxe-server/#chap-installation-server-setup.

[12] J. Salter, «ZFS 101—Understanding ZFS storage and performance», *Ars Technica*, ago. 05, 2020. https://arstechnica.com/information-technology/2020/05/zfs-101-understanding-zfs-storage-and-performance/.

[13] «Chapter 19. The Z File System (ZFS)». https://www.freebsd.org/doc/handbook/zfs.html.

[14] «zfsonlinux/pkg-zfs», *GitHub*. https://github.com/zfsonlinux/pkg-zfs.

[15] «Chapter 2. Using Active Directory as an Identity Provider for SSSD Red Hat Enterprise Linux 7», *Red Hat Customer Portal*. https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/windows_integration_guide/sssd-ad.

[16] «Chapter 3. Using realmd to Connect to an Active Directory Domain Red Hat Enterprise Linux 7», *Red Hat Customer Portal*. https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/windows_integration_guide/ch-configuring_authentication

[17] «8.3. autofs Red Hat Enterprise Linux 7», *Red Hat Customer Portal*. https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/storage_administration_guide/nfs-autofs

[18] «LustreArchitecture-v4.pdf». [En línea]. Disponible en: http://wiki.lustre.org/images/6/64/LustreArchitecture-v4.pdf.

[19] «Lustre Wiki». http://wiki.lustre.org/Main_Page.

[20] «Lustre* Software Release 2.x - Operations Manual», p. 598.

# Anexo A: Primer script de instalación del Sistema de ficheros raíz ZFS

```
#!/bin/bash
#-x

#lsblk -o NAME,MODEL
DISK2=$1

################ Required Software ###############
yum update -y
yum install -y gdisk rsync efibootmgr
yum install -y grub2-efi-x64-modules shim #modules in /usr/lib/grub/x86_64-
efi
yum install -y http://download.zfsonlinux.org/epel/zfs-re-
lease.el7_6.noarch.rpm

#edit /etc/yum.repos.d/zfs.repo ###[zfs-kmod]+enabled=1 [zfs]=+enabled=0
rm  /etc/yum.repos.d/zfs.repo
cp  /tmp/zfs.repo /etc/yum.repos.d/

yum install -y zfs
yum install -y zfs-dracut

################ DISK2 Partitioning ################
#Reset partition table::::
sgdisk -o $DISK2
sgdisk -g $DISK2

#UEFI boot partition
sgdisk -n 8:-500M $DISK2
sgdisk -t 8:ef00 $DISK2
sgdisk -c 8:"EFI BOOT partition" $DISK2

#zfs partition
sgdisk -n 1: $DISK2
sgdisk -t 1:bf01 $DISK2
sgdisk -c 1:"Solaris /usr & Mac ZFS" $DISK2

################ Create zpool ####################

sleep 10 #Time needed after partitioning for the new links in /dev

wipefs -a "$DISK2"-part1 #In case theres an old fs signature after creating
the partiton
wipefs -a "$DISK2"-part8 #In case theres an old fs signature after creating
the partiton

#Remove posible zpool previously using the Disk
zpool labelclear -f "$DISK2"-part1 # frees the partition from any pool; op-
tion -f: treats exported pools as inactive

zpool create -d -o feature@async_destroy=enabled -o feature@empty_bpobj=en-
abled -o feature@lz4_compress=enabled -o ashift=12 -O compression=lz4 rpool
"$DISK2"-part1 &> create.out #debug; option -f if already exists an UFS
filesystem on the partition


#afterwards to make sure that the new udev rule is run.
```

```
udevadm trigger

zfs create rpool/ROOT

############### rsync root filesystem DISK1 --> DISK2 ##############

mkdir /mnt/tmp

#copy  /
mount --bind / /mnt/tmp
rsync -aqPX /mnt/tmp/. /rpool/ROOT/.
umount /mnt/tmp

#add other filesystems to copy if needed

############### Edit some configuration files on Disk2 #############

export ZPOOL_VDEV_NAME_PATH=YES

#edit fstab: Remove the original partitions
rm /rpool/ROOT/etc/fstab
cp /tmp/fstab /rpool/ROOT/etc/

#edit default grub
#boot=zfs root=ZFS=rpool/ROOT
#GRUB_PRELOAD_MODULES="part_gpt zfs"
rm /rpool/ROOT/etc/default/grub
cp /tmp/grub /rpool/ROOT/etc/default/

############### Rebuild initramfs on Disk2 ######################

for dir in proc sys dev;do mount --bind /$dir /rpool/ROOT/$dir;done

#edit /etc/dracut.conf
#add_dracutmodules+="zfs"
rm /rpool/ROOT/etc/dracut.conf
cp /tmp/dracut.conf /rpool/ROOT/etc/

chroot /rpool/ROOT zgenhostid $(hostid) &> zgen.out #Avoid needing to force
import, adding the hostid to the initramfs

# zfs-import-cache wait 30 seconds: needed with the kernel 3.10.0-957
#chroot /rpool/ROOT rm /usr/lib/systemd/system/zfs-import-cache.service
#chroot /rpool/ROOT cp /tmp/zfs-import-cache_wait /usr/lib/systemd/sys-
tem/zfs-import-cache.service

chroot /rpool/ROOT/ dracut -f /boot/initramfs-$(uname -r).img $(uname -r)
#kernel despues del update
#chroot /rpool/ROOT/ dracut -f /boot/initramfs-3.10.0-957.el7.x86_64.img
3.10.0-957.el7.x86_64 &> dracut1.out #debug; option -f: override existing
file
#chroot /rpool/ROOT/ dracut -f /boot/initramfs-3.10.0-
957.27.2.el7.x86_64.img 3.10.0-957.27.2.el7.x86_64 &> dracut2.out #debug;
option -f: override existing file

################ Install GRUB on DISK2 ####################

# Format the UEFI boot partition
mkfs.fat -F32 -s 1 "$DISK2"-part8

# Copy UEFI boot partition content from the original installation
```

```
mkdir /mnt/tmp
mount "$DISK2"-part8 /mnt/tmp
cp -rf /boot/efi/EFI /mnt/tmp/
cp -rf /usr/lib/grub/x86_64-efi/ /mnt/tmp/EFI/centos/ #Add grub-efi modules
that contains the zfs module

# Create new grub.cfg
rm /mnt/tmp/EFI/centos/grub.cfg
chroot /rpool/ROOT/ grub2-mkconfig --output=/tmp/grub.cfg    #BIOS
mv /rpool/ROOT/tmp/grub.cfg /mnt/tmp/EFI/centos/
umount /mnt/tmp

for dir in proc sys dev;do umount  /rpool/ROOT/$dir;done

#################### Update bootorder ######################

# Creates a new boot-menu entry
efibootmgr -c -d $DISK2 -p 8 -L "CentOS DISK 2" -l "/EFI/CEN-
TOS/GRUBX64.EFI"

INDEX2="$(efibootmgr -v | grep "DISK 2" | tail -1 | cut -c5-8)" # The new
entry takes the first spot in the bootorder by default

efibootmgr -n $INDEX2
exit
# Reboot by Ansible
```

# Anexo B: Segundo script de instalación del Sistema de ficheros raíz ZFS

```
#!/bin/bash
#-x

DISK1=$1
DISK2=$2

################## DISK1 Partitioning ####################

#It is necessary to erase the filesystem of the partition that will form
the mirror zfs, if not, even if the disk is partitioned again there are
still remnants of the filesystem.
wipefs -a "$DISK1"-part1

#clear the partition table
sgdisk -o $DISK1
sgdisk -g $DISK1

##UEFI boot partition
sgdisk -n 8:-500M $DISK1
sgdisk -t 8:ef00 $DISK1
sgdisk -c 8:"EFI BOOT partition" $DISK1
##zfs partition
sgdisk -n 1: $DISK1
sgdisk -t 1:bf01 $DISK1
sgdisk -c 1:"Solaris /usr & Mac ZFS" $DISK1

sleep 10 #Time needed after partitioning for the new links in /dev


################### Create mirror with Disk2 and Disk1 ################

zpool import &> import1_zfs2.out #debug
zpool labelclear -f "$DISK1"-part1 # frees the partition from any pool; op-
tion -f: treats exported pools as inactive
zpool import &> import2_zfs2.out #debug
zpool attach rpool "$DISK2"-part1 "$DISK1"-part1 &> mirror.out #zpool at-
tach creates a mirror with both disks
udevadm trigger
zpool import &> import3_zfs2.out #debug

#wait for resilvering (estimated time)
sleep 240 #Quitar


################ Rebuild initramfs ####################

#Rebuild initramfs to update the zfs cache file
zpool set cachefile=/etc/zfs/zpool.cache rpool
dracut -f /boot/initramfs-$(uname -r).img $(uname -r)
#dracut -f /boot/initramfs-3.10.0-957.el7.x86_64.img 3.10.0-957.el7.x86_64

#Wait time is not needed once the root filesystem is mounted (is only
needed in the initramfs service)
#rm /usr/lib/systemd/system/zfs-import-cache.service
#cp /tmp/zfs-import-cache /usr/lib/systemd/system/zfs-import-cache.service
```

```
################## Install GRUB on DISK1 ###################

mount "$DISK2"-part8 /boot/efi #BIOS
mkfs.fat -F32 -s 1 "$DISK1"-part8
mount "$DISK1"-part8 /mnt/tmp
cp -rf /boot/efi/EFI /mnt/tmp


umount /boot/efi
umount /mnt/tmp

#################### Update boot-menu ######################

efibootmgr -c -d $DISK1 -p 8 -L "CentOS DISK 1" -l "/EFI/CEN-
TOS/GRUBX64.EFI"

INDEX1="$(efibootmgr -v | grep "DISK 1" | head -1 | cut -c5-8)"
INDEX2="$(efibootmgr -v | grep "DISK 2" | head -1 | cut -c5-8)"

efibootmgr -o "$INDEX1","$INDEX2"

efibootmgr -v &> efi1.out
exit
# Reboot by Ansible
```